| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ConvolveOp.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/image/ComponentSampleModel.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/image/CropImageFilter.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/image/ConvolveOp.html)    [**NO FRAMES**](http://docs.google.com/ConvolveOp.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#17dp8vu) | [METHOD](#lnxbz9) |

## **java.awt.image**

Class ConvolveOp

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

**All Implemented Interfaces:** [BufferedImageOp](http://docs.google.com/java/awt/image/BufferedImageOp.html), [RasterOp](http://docs.google.com/java/awt/image/RasterOp.html)

public class **ConvolveOp**extends [Object](http://docs.google.com/java/lang/Object.html)implements [BufferedImageOp](http://docs.google.com/java/awt/image/BufferedImageOp.html), [RasterOp](http://docs.google.com/java/awt/image/RasterOp.html)

This class implements a convolution from the source to the destination. Convolution using a convolution kernel is a spatial operation that computes the output pixel from an input pixel by multiplying the kernel with the surround of the input pixel. This allows the output pixel to be affected by the immediate neighborhood in a way that can be mathematically specified with a kernel.

This class operates with BufferedImage data in which color components are premultiplied with the alpha component. If the Source BufferedImage has an alpha component, and the color components are not premultiplied with the alpha component, then the data are premultiplied before being convolved. If the Destination has color components which are not premultiplied, then alpha is divided out before storing into the Destination (if alpha is 0, the color components are set to 0). If the Destination has no alpha component, then the resulting alpha is discarded after first dividing it out of the color components.

Rasters are treated as having no alpha channel. If the above treatment of the alpha channel in BufferedImages is not desired, it may be avoided by getting the Raster of a source BufferedImage and using the filter method of this class which works with Rasters.

If a RenderingHints object is specified in the constructor, the color rendering hint and the dithering hint may be used when color conversion is required.

Note that the Source and the Destination may not be the same object.

**See Also:**[Kernel](http://docs.google.com/java/awt/image/Kernel.html), [RenderingHints.KEY\_COLOR\_RENDERING](http://docs.google.com/java/awt/RenderingHints.html#KEY_COLOR_RENDERING), [RenderingHints.KEY\_DITHERING](http://docs.google.com/java/awt/RenderingHints.html#KEY_DITHERING)

| **Field Summary** | |
| --- | --- |
| static int | [**EDGE\_NO\_OP**](http://docs.google.com/java/awt/image/ConvolveOp.html#EDGE_NO_OP)            Pixels at the edge of the source image are copied to the corresponding pixels in the destination without modification. |
| static int | [**EDGE\_ZERO\_FILL**](http://docs.google.com/java/awt/image/ConvolveOp.html#EDGE_ZERO_FILL)            Pixels at the edge of the destination image are set to zero. |

| **Constructor Summary** | |
| --- | --- |
| [**ConvolveOp**](http://docs.google.com/java/awt/image/ConvolveOp.html#ConvolveOp(java.awt.image.Kernel))([Kernel](http://docs.google.com/java/awt/image/Kernel.html) kernel)            Constructs a ConvolveOp given a Kernel. |
| [**ConvolveOp**](http://docs.google.com/java/awt/image/ConvolveOp.html#ConvolveOp(java.awt.image.Kernel,%20int,%20java.awt.RenderingHints))([Kernel](http://docs.google.com/java/awt/image/Kernel.html) kernel, int edgeCondition, [RenderingHints](http://docs.google.com/java/awt/RenderingHints.html) hints)            Constructs a ConvolveOp given a Kernel, an edge condition, and a RenderingHints object (which may be null). |

| **Method Summary** | |
| --- | --- |
| [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) | [**createCompatibleDestImage**](http://docs.google.com/java/awt/image/ConvolveOp.html#createCompatibleDestImage(java.awt.image.BufferedImage,%20java.awt.image.ColorModel))([BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) src, [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) destCM)            Creates a zeroed destination image with the correct size and number of bands. |
| [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) | [**createCompatibleDestRaster**](http://docs.google.com/java/awt/image/ConvolveOp.html#createCompatibleDestRaster(java.awt.image.Raster))([Raster](http://docs.google.com/java/awt/image/Raster.html) src)            Creates a zeroed destination Raster with the correct size and number of bands, given this source. |
| [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) | [**filter**](http://docs.google.com/java/awt/image/ConvolveOp.html#filter(java.awt.image.BufferedImage,%20java.awt.image.BufferedImage))([BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) src, [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) dst)            Performs a convolution on BufferedImages. |
| [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) | [**filter**](http://docs.google.com/java/awt/image/ConvolveOp.html#filter(java.awt.image.Raster,%20java.awt.image.WritableRaster))([Raster](http://docs.google.com/java/awt/image/Raster.html) src, [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) dst)            Performs a convolution on Rasters. |
| [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) | [**getBounds2D**](http://docs.google.com/java/awt/image/ConvolveOp.html#getBounds2D(java.awt.image.BufferedImage))([BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) src)            Returns the bounding box of the filtered destination image. |
| [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) | [**getBounds2D**](http://docs.google.com/java/awt/image/ConvolveOp.html#getBounds2D(java.awt.image.Raster))([Raster](http://docs.google.com/java/awt/image/Raster.html) src)            Returns the bounding box of the filtered destination Raster. |
| int | [**getEdgeCondition**](http://docs.google.com/java/awt/image/ConvolveOp.html#getEdgeCondition())()            Returns the edge condition. |
| [Kernel](http://docs.google.com/java/awt/image/Kernel.html) | [**getKernel**](http://docs.google.com/java/awt/image/ConvolveOp.html#getKernel())()            Returns the Kernel. |
| [Point2D](http://docs.google.com/java/awt/geom/Point2D.html) | [**getPoint2D**](http://docs.google.com/java/awt/image/ConvolveOp.html#getPoint2D(java.awt.geom.Point2D,%20java.awt.geom.Point2D))([Point2D](http://docs.google.com/java/awt/geom/Point2D.html) srcPt, [Point2D](http://docs.google.com/java/awt/geom/Point2D.html) dstPt)            Returns the location of the destination point given a point in the source. |
| [RenderingHints](http://docs.google.com/java/awt/RenderingHints.html) | [**getRenderingHints**](http://docs.google.com/java/awt/image/ConvolveOp.html#getRenderingHints())()            Returns the rendering hints for this op. |

| **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** |
| --- |

### EDGE\_ZERO\_FILL

public static final int **EDGE\_ZERO\_FILL**

Pixels at the edge of the destination image are set to zero. This is the default.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.image.ConvolveOp.EDGE_ZERO_FILL)

### EDGE\_NO\_OP

public static final int **EDGE\_NO\_OP**

Pixels at the edge of the source image are copied to the corresponding pixels in the destination without modification.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.image.ConvolveOp.EDGE_NO_OP)

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

### ConvolveOp

public **ConvolveOp**([Kernel](http://docs.google.com/java/awt/image/Kernel.html) kernel,  
 int edgeCondition,  
 [RenderingHints](http://docs.google.com/java/awt/RenderingHints.html) hints)

Constructs a ConvolveOp given a Kernel, an edge condition, and a RenderingHints object (which may be null).

**Parameters:**kernel - the specified KerneledgeCondition - the specified edge conditionhints - the specified RenderingHints object**See Also:**[Kernel](http://docs.google.com/java/awt/image/Kernel.html), [EDGE\_NO\_OP](http://docs.google.com/java/awt/image/ConvolveOp.html#EDGE_NO_OP), [EDGE\_ZERO\_FILL](http://docs.google.com/java/awt/image/ConvolveOp.html#EDGE_ZERO_FILL), [RenderingHints](http://docs.google.com/java/awt/RenderingHints.html)

### ConvolveOp

public **ConvolveOp**([Kernel](http://docs.google.com/java/awt/image/Kernel.html) kernel)

Constructs a ConvolveOp given a Kernel. The edge condition will be EDGE\_ZERO\_FILL.

**Parameters:**kernel - the specified Kernel**See Also:**[Kernel](http://docs.google.com/java/awt/image/Kernel.html), [EDGE\_ZERO\_FILL](http://docs.google.com/java/awt/image/ConvolveOp.html#EDGE_ZERO_FILL)

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

### getEdgeCondition

public int **getEdgeCondition**()

Returns the edge condition.

**Returns:**the edge condition of this ConvolveOp.**See Also:**[EDGE\_NO\_OP](http://docs.google.com/java/awt/image/ConvolveOp.html#EDGE_NO_OP), [EDGE\_ZERO\_FILL](http://docs.google.com/java/awt/image/ConvolveOp.html#EDGE_ZERO_FILL)

### getKernel

public final [Kernel](http://docs.google.com/java/awt/image/Kernel.html) **getKernel**()

Returns the Kernel.

**Returns:**the Kernel of this ConvolveOp.

### filter

public final [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) **filter**([BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) src,  
 [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) dst)

Performs a convolution on BufferedImages. Each component of the source image will be convolved (including the alpha component, if present). If the color model in the source image is not the same as that in the destination image, the pixels will be converted in the destination. If the destination image is null, a BufferedImage will be created with the source ColorModel. The IllegalArgumentException may be thrown if the source is the same as the destination.

**Specified by:**[filter](http://docs.google.com/java/awt/image/BufferedImageOp.html#filter(java.awt.image.BufferedImage,%20java.awt.image.BufferedImage)) in interface [BufferedImageOp](http://docs.google.com/java/awt/image/BufferedImageOp.html) **Parameters:**src - the source BufferedImage to filterdst - the destination BufferedImage for the filtered src **Returns:**the filtered BufferedImage **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if src is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if src equals dst [ImagingOpException](http://docs.google.com/java/awt/image/ImagingOpException.html) - if src cannot be filtered

### filter

public final [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) **filter**([Raster](http://docs.google.com/java/awt/image/Raster.html) src,  
 [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) dst)

Performs a convolution on Rasters. Each band of the source Raster will be convolved. The source and destination must have the same number of bands. If the destination Raster is null, a new Raster will be created. The IllegalArgumentException may be thrown if the source is the same as the destination.

**Specified by:**[filter](http://docs.google.com/java/awt/image/RasterOp.html#filter(java.awt.image.Raster,%20java.awt.image.WritableRaster)) in interface [RasterOp](http://docs.google.com/java/awt/image/RasterOp.html) **Parameters:**src - the source Raster to filterdst - the destination WritableRaster for the filtered src **Returns:**the filtered WritableRaster **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if src is null [ImagingOpException](http://docs.google.com/java/awt/image/ImagingOpException.html) - if src and dst do not have the same number of bands [ImagingOpException](http://docs.google.com/java/awt/image/ImagingOpException.html) - if src cannot be filtered [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if src equals dst

### createCompatibleDestImage

public [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) **createCompatibleDestImage**([BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) src,  
 [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) destCM)

Creates a zeroed destination image with the correct size and number of bands. If destCM is null, an appropriate ColorModel will be used.

**Specified by:**[createCompatibleDestImage](http://docs.google.com/java/awt/image/BufferedImageOp.html#createCompatibleDestImage(java.awt.image.BufferedImage,%20java.awt.image.ColorModel)) in interface [BufferedImageOp](http://docs.google.com/java/awt/image/BufferedImageOp.html) **Parameters:**src - Source image for the filter operation.destCM - ColorModel of the destination. Can be null. **Returns:**a destination BufferedImage with the correct size and number of bands.

### createCompatibleDestRaster

public [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) **createCompatibleDestRaster**([Raster](http://docs.google.com/java/awt/image/Raster.html) src)

Creates a zeroed destination Raster with the correct size and number of bands, given this source.

**Specified by:**[createCompatibleDestRaster](http://docs.google.com/java/awt/image/RasterOp.html#createCompatibleDestRaster(java.awt.image.Raster)) in interface [RasterOp](http://docs.google.com/java/awt/image/RasterOp.html) **Parameters:**src - the source Raster **Returns:**a WritableRaster that is compatible with src

### getBounds2D

public final [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) **getBounds2D**([BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) src)

Returns the bounding box of the filtered destination image. Since this is not a geometric operation, the bounding box does not change.

**Specified by:**[getBounds2D](http://docs.google.com/java/awt/image/BufferedImageOp.html#getBounds2D(java.awt.image.BufferedImage)) in interface [BufferedImageOp](http://docs.google.com/java/awt/image/BufferedImageOp.html) **Parameters:**src - The BufferedImage to be filtered **Returns:**The Rectangle2D representing the destination image's bounding box.

### getBounds2D

public final [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) **getBounds2D**([Raster](http://docs.google.com/java/awt/image/Raster.html) src)

Returns the bounding box of the filtered destination Raster. Since this is not a geometric operation, the bounding box does not change.

**Specified by:**[getBounds2D](http://docs.google.com/java/awt/image/RasterOp.html#getBounds2D(java.awt.image.Raster)) in interface [RasterOp](http://docs.google.com/java/awt/image/RasterOp.html) **Parameters:**src - the source Raster **Returns:**a Rectangle2D that is the bounding box of the Raster resulting from the filtering operation.

### getPoint2D

public final [Point2D](http://docs.google.com/java/awt/geom/Point2D.html) **getPoint2D**([Point2D](http://docs.google.com/java/awt/geom/Point2D.html) srcPt,  
 [Point2D](http://docs.google.com/java/awt/geom/Point2D.html) dstPt)

Returns the location of the destination point given a point in the source. If dstPt is non-null, it will be used to hold the return value. Since this is not a geometric operation, the srcPt will equal the dstPt.

**Specified by:**[getPoint2D](http://docs.google.com/java/awt/image/BufferedImageOp.html#getPoint2D(java.awt.geom.Point2D,%20java.awt.geom.Point2D)) in interface [BufferedImageOp](http://docs.google.com/java/awt/image/BufferedImageOp.html)**Specified by:**[getPoint2D](http://docs.google.com/java/awt/image/RasterOp.html#getPoint2D(java.awt.geom.Point2D,%20java.awt.geom.Point2D)) in interface [RasterOp](http://docs.google.com/java/awt/image/RasterOp.html) **Parameters:**srcPt - the Point2D that represents the point in the source imagedstPt - The Point2D in which to store the result **Returns:**The Point2D in the destination image that corresponds to the specified point in the source image.

### getRenderingHints

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

Returns the rendering hints for this op.

**Specified by:**[getRenderingHints](http://docs.google.com/java/awt/image/BufferedImageOp.html#getRenderingHints()) in interface [BufferedImageOp](http://docs.google.com/java/awt/image/BufferedImageOp.html)**Specified by:**[getRenderingHints](http://docs.google.com/java/awt/image/RasterOp.html#getRenderingHints()) in interface [RasterOp](http://docs.google.com/java/awt/image/RasterOp.html) **Returns:**The RenderingHints object for this BufferedImageOp. Returns null if no hints have been set.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ConvolveOp.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/image/ComponentSampleModel.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/image/CropImageFilter.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/image/ConvolveOp.html)    [**NO FRAMES**](http://docs.google.com/ConvolveOp.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#17dp8vu) | [METHOD](#lnxbz9) |

[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).