| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AffineTransformOp.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   [**NEXT CLASS**](http://docs.google.com/java/awt/image/AreaAveragingScaleFilter.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/image/AffineTransformOp.html)    [**NO FRAMES**](http://docs.google.com/AffineTransformOp.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#3rdcrjn) | [METHOD](#35nkun2) |

## **java.awt.image**

Class AffineTransformOp

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

**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 **AffineTransformOp**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 uses an affine transform to perform a linear mapping from 2D coordinates in the source image or Raster to 2D coordinates in the destination image or Raster. The type of interpolation that is used is specified through a constructor, either by a RenderingHints object or by one of the integer interpolation types defined in this class.

If a RenderingHints object is specified in the constructor, the interpolation hint and the rendering quality hint are used to set the interpolation type for this operation. The color rendering hint and the dithering hint can be used when color conversion is required.

Note that the following constraints have to be met:

* The source and destination must be different.
* For Raster objects, the number of bands in the source must be equal to the number of bands in the destination.

**See Also:**[AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html), [BufferedImageFilter](http://docs.google.com/java/awt/image/BufferedImageFilter.html), [RenderingHints.KEY\_INTERPOLATION](http://docs.google.com/java/awt/RenderingHints.html#KEY_INTERPOLATION), [RenderingHints.KEY\_RENDERING](http://docs.google.com/java/awt/RenderingHints.html#KEY_RENDERING), [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 | [**TYPE\_BICUBIC**](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_BICUBIC)            Bicubic interpolation type. |
| static int | [**TYPE\_BILINEAR**](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_BILINEAR)            Bilinear interpolation type. |
| static int | [**TYPE\_NEAREST\_NEIGHBOR**](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_NEAREST_NEIGHBOR)            Nearest-neighbor interpolation type. |

| **Constructor Summary** | |
| --- | --- |
| [**AffineTransformOp**](http://docs.google.com/java/awt/image/AffineTransformOp.html#AffineTransformOp(java.awt.geom.AffineTransform,%20int))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) xform, int interpolationType)            Constructs an AffineTransformOp given an affine transform and the interpolation type. |
| [**AffineTransformOp**](http://docs.google.com/java/awt/image/AffineTransformOp.html#AffineTransformOp(java.awt.geom.AffineTransform,%20java.awt.RenderingHints))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) xform, [RenderingHints](http://docs.google.com/java/awt/RenderingHints.html) hints)            Constructs an AffineTransformOp given an affine transform. |

| **Method Summary** | |
| --- | --- |
| [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) | [**createCompatibleDestImage**](http://docs.google.com/java/awt/image/AffineTransformOp.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/AffineTransformOp.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. |
| [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) | [**filter**](http://docs.google.com/java/awt/image/AffineTransformOp.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)            Transforms the source BufferedImage and stores the results in the destination BufferedImage. |
| [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) | [**filter**](http://docs.google.com/java/awt/image/AffineTransformOp.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)            Transforms the source Raster and stores the results in the destination Raster. |
| [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) | [**getBounds2D**](http://docs.google.com/java/awt/image/AffineTransformOp.html#getBounds2D(java.awt.image.BufferedImage))([BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) src)            Returns the bounding box of the transformed destination. |
| [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) | [**getBounds2D**](http://docs.google.com/java/awt/image/AffineTransformOp.html#getBounds2D(java.awt.image.Raster))([Raster](http://docs.google.com/java/awt/image/Raster.html) src)            Returns the bounding box of the transformed destination. |
| int | [**getInterpolationType**](http://docs.google.com/java/awt/image/AffineTransformOp.html#getInterpolationType())()            Returns the interpolation type used by this op. |
| [Point2D](http://docs.google.com/java/awt/geom/Point2D.html) | [**getPoint2D**](http://docs.google.com/java/awt/image/AffineTransformOp.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 corresponding 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/AffineTransformOp.html#getRenderingHints())()            Returns the rendering hints used by this transform operation. |
| [AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) | [**getTransform**](http://docs.google.com/java/awt/image/AffineTransformOp.html#getTransform())()            Returns the affine transform used by this transform operation. |

| **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** |
| --- |

### TYPE\_NEAREST\_NEIGHBOR

public static final int **TYPE\_NEAREST\_NEIGHBOR**

Nearest-neighbor interpolation type.

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

### TYPE\_BILINEAR

public static final int **TYPE\_BILINEAR**

Bilinear interpolation type.

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

### TYPE\_BICUBIC

public static final int **TYPE\_BICUBIC**

Bicubic interpolation type.

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

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

### AffineTransformOp

public **AffineTransformOp**([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) xform,  
 [RenderingHints](http://docs.google.com/java/awt/RenderingHints.html) hints)

Constructs an AffineTransformOp given an affine transform. The interpolation type is determined from the RenderingHints object. If the interpolation hint is defined, it will be used. Otherwise, if the rendering quality hint is defined, the interpolation type is determined from its value. If no hints are specified (hints is null), the interpolation type is [TYPE\_NEAREST\_NEIGHBOR](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_NEAREST_NEIGHBOR).

**Parameters:**xform - The AffineTransform to use for the operation.hints - The RenderingHints object used to specify the interpolation type for the operation. **Throws:** [ImagingOpException](http://docs.google.com/java/awt/image/ImagingOpException.html) - if the transform is non-invertible.**See Also:**[RenderingHints.KEY\_INTERPOLATION](http://docs.google.com/java/awt/RenderingHints.html#KEY_INTERPOLATION), [RenderingHints.KEY\_RENDERING](http://docs.google.com/java/awt/RenderingHints.html#KEY_RENDERING)

### AffineTransformOp

public **AffineTransformOp**([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) xform,  
 int interpolationType)

Constructs an AffineTransformOp given an affine transform and the interpolation type.

**Parameters:**xform - The AffineTransform to use for the operation.interpolationType - One of the integer interpolation type constants defined by this class: [TYPE\_NEAREST\_NEIGHBOR](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_NEAREST_NEIGHBOR), [TYPE\_BILINEAR](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_BILINEAR), [TYPE\_BICUBIC](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_BICUBIC). **Throws:** [ImagingOpException](http://docs.google.com/java/awt/image/ImagingOpException.html) - if the transform is non-invertible.

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

### getInterpolationType

public final int **getInterpolationType**()

Returns the interpolation type used by this op.

**Returns:**the interpolation type.**See Also:**[TYPE\_NEAREST\_NEIGHBOR](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_NEAREST_NEIGHBOR), [TYPE\_BILINEAR](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_BILINEAR), [TYPE\_BICUBIC](http://docs.google.com/java/awt/image/AffineTransformOp.html#TYPE_BICUBIC)

### 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)

Transforms the source BufferedImage and stores the results in the destination BufferedImage. If the color models for the two images do not match, a color conversion into the destination color model is performed. If the destination image is null, a BufferedImage is created with the source ColorModel.

The coordinates of the rectangle returned by getBounds2D(BufferedImage) are not necessarily the same as the coordinates of the BufferedImage returned by this method. If the upper-left corner coordinates of the rectangle are negative then this part of the rectangle is not drawn. If the upper-left corner coordinates of the rectangle are positive then the filtered image is drawn at that position in the destination BufferedImage.

An IllegalArgumentException is 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 BufferedImage to transform.dst - The BufferedImage in which to store the results of the transformation. **Returns:**The filtered BufferedImage. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if src and dst are the same [ImagingOpException](http://docs.google.com/java/awt/image/ImagingOpException.html) - if the image cannot be transformed because of a data-processing error that might be caused by an invalid image format, tile format, or image-processing operation, or any other unsupported operation.

### 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)

Transforms the source Raster and stores the results in the destination Raster. This operation performs the transform band by band.

If the destination Raster is null, a new Raster is created. An IllegalArgumentException may be thrown if the source is the same as the destination or if the number of bands in the source is not equal to the number of bands in the destination.

The coordinates of the rectangle returned by getBounds2D(Raster) are not necessarily the same as the coordinates of the WritableRaster returned by this method. If the upper-left corner coordinates of rectangle are negative then this part of the rectangle is not drawn. If the coordinates of the rectangle are positive then the filtered image is drawn at that position in the destination Raster.

**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 Raster to transform.dst - The Raster in which to store the results of the transformation. **Returns:**The transformed Raster. **Throws:** [ImagingOpException](http://docs.google.com/java/awt/image/ImagingOpException.html) - if the raster cannot be transformed because of a data-processing error that might be caused by an invalid image format, tile format, or image-processing operation, or any other unsupported operation.

### 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 transformed destination. The rectangle returned is the actual bounding box of the transformed points. The coordinates of the upper-left corner of the returned rectangle might not be (0, 0).

**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 transformed. **Returns:**The Rectangle2D representing the destination'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 transformed destination. The rectangle returned will be the actual bounding box of the transformed points. The coordinates of the upper-left corner of the returned rectangle might not be (0, 0).

**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 Raster to be transformed. **Returns:**The Rectangle2D representing the destination's bounding box.

### 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. A RasterFormatException may be thrown if the transformed width or height is equal to 0.

If destCM is null, an appropriate ColorModel is used; this ColorModel may have an alpha channel even if the source ColorModel is opaque.

**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 - The BufferedImage to be transformed.destCM - ColorModel of the destination. If null, an appropriate ColorModel is used. **Returns:**The zeroed destination image.

### 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. A RasterFormatException may be thrown if the transformed width or height is equal to 0.

**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 Raster to be transformed. **Returns:**The zeroed destination Raster.

### 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 corresponding destination point given a point in the source. If dstPt is specified, it is used to hold the return value.

**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 source point.dstPt - The Point2D in which to store the result. **Returns:**The Point2D in the destination that corresponds to the specified point in the source.

### getTransform

public final [AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) **getTransform**()

Returns the affine transform used by this transform operation.

**Returns:**The AffineTransform associated with this op.

### getRenderingHints

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

Returns the rendering hints used by this transform operation.

**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 associated with this op.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AffineTransformOp.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   [**NEXT CLASS**](http://docs.google.com/java/awt/image/AreaAveragingScaleFilter.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/image/AffineTransformOp.html)    [**NO FRAMES**](http://docs.google.com/AffineTransformOp.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#3rdcrjn) | [METHOD](#35nkun2) |

[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).