| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/IIOParam.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/imageio/IIOImage.html)   [**NEXT CLASS**](http://docs.google.com/javax/imageio/IIOParamController.html) | [**FRAMES**](http://docs.google.com/index.html?javax/imageio/IIOParam.html)    [**NO FRAMES**](http://docs.google.com/IIOParam.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#z337ya) | [METHOD](#1y810tw) |

## **javax.imageio**

Class IIOParam

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

**Direct Known Subclasses:** [ImageReadParam](http://docs.google.com/javax/imageio/ImageReadParam.html), [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html)

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

A superclass of all classes describing how streams should be decoded or encoded. This class contains all the variables and methods that are shared by ImageReadParam and ImageWriteParam.

This class provides mechanisms to specify a source region and a destination region. When reading, the source is the stream and the in-memory image is the destination. When writing, these are reversed. In the case of writing, destination regions may be used only with a writer that supports pixel replacement.

Decimation subsampling may be specified for both readers and writers, using a movable subsampling grid.

Subsets of the source and destination bands may be selected.

| **Field Summary** | |
| --- | --- |
| protected  [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) | [**controller**](http://docs.google.com/javax/imageio/IIOParam.html#controller)            The IIOParamController that will be used to provide settings for this IIOParam object when the activateController method is called. |
| protected  [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) | [**defaultController**](http://docs.google.com/javax/imageio/IIOParam.html#defaultController)            The default IIOParamController that will be used to provide settings for this IIOParam object when the activateController method is called. |
| protected  [Point](http://docs.google.com/java/awt/Point.html) | [**destinationOffset**](http://docs.google.com/javax/imageio/IIOParam.html#destinationOffset)            The offset in the destination where the upper-left decoded pixel should be placed. |
| protected  [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) | [**destinationType**](http://docs.google.com/javax/imageio/IIOParam.html#destinationType)            An ImageTypeSpecifier to be used to generate a destination image when reading, or to set the output color type when writing. |
| protected  int[] | [**sourceBands**](http://docs.google.com/javax/imageio/IIOParam.html#sourceBands)            An array of ints indicating which source bands will be used, or null. |
| protected  [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**sourceRegion**](http://docs.google.com/javax/imageio/IIOParam.html#sourceRegion)            The source region, on null if none is set. |
| protected  int | [**sourceXSubsampling**](http://docs.google.com/javax/imageio/IIOParam.html#sourceXSubsampling)            The decimation subsampling to be applied in the horizontal direction. |
| protected  int | [**sourceYSubsampling**](http://docs.google.com/javax/imageio/IIOParam.html#sourceYSubsampling)            The decimation subsampling to be applied in the vertical direction. |
| protected  int | [**subsamplingXOffset**](http://docs.google.com/javax/imageio/IIOParam.html#subsamplingXOffset)            A horizontal offset to be applied to the subsampling grid before subsampling. |
| protected  int | [**subsamplingYOffset**](http://docs.google.com/javax/imageio/IIOParam.html#subsamplingYOffset)            A vertical offset to be applied to the subsampling grid before subsampling. |

| **Constructor Summary** | |
| --- | --- |
| protected | [**IIOParam**](http://docs.google.com/javax/imageio/IIOParam.html#IIOParam())()            Protected constructor may be called only by subclasses. |

| **Method Summary** | |
| --- | --- |
| boolean | [**activateController**](http://docs.google.com/javax/imageio/IIOParam.html#activateController())()            Activates the installed IIOParamController for this IIOParam object and returns the resulting value. |
| [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) | [**getController**](http://docs.google.com/javax/imageio/IIOParam.html#getController())()            Returns whatever IIOParamController is currently installed. |
| [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) | [**getDefaultController**](http://docs.google.com/javax/imageio/IIOParam.html#getDefaultController())()            Returns the default IIOParamController, if there is one, regardless of the currently installed controller. |
| [Point](http://docs.google.com/java/awt/Point.html) | [**getDestinationOffset**](http://docs.google.com/javax/imageio/IIOParam.html#getDestinationOffset())()            Returns the offset in the destination image at which pixels are to be placed. |
| [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) | [**getDestinationType**](http://docs.google.com/javax/imageio/IIOParam.html#getDestinationType())()            Returns the type of image to be returned by the read, if one was set by a call to setDestination(ImageTypeSpecifier), as an ImageTypeSpecifier. |
| int[] | [**getSourceBands**](http://docs.google.com/javax/imageio/IIOParam.html#getSourceBands())()            Returns the set of of source bands to be used. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getSourceRegion**](http://docs.google.com/javax/imageio/IIOParam.html#getSourceRegion())()            Returns the source region to be used. |
| int | [**getSourceXSubsampling**](http://docs.google.com/javax/imageio/IIOParam.html#getSourceXSubsampling())()            Returns the number of source columns to advance for each pixel. |
| int | [**getSourceYSubsampling**](http://docs.google.com/javax/imageio/IIOParam.html#getSourceYSubsampling())()            Returns the number of rows to advance for each pixel. |
| int | [**getSubsamplingXOffset**](http://docs.google.com/javax/imageio/IIOParam.html#getSubsamplingXOffset())()            Returns the horizontal offset of the subsampling grid. |
| int | [**getSubsamplingYOffset**](http://docs.google.com/javax/imageio/IIOParam.html#getSubsamplingYOffset())()            Returns the vertical offset of the subsampling grid. |
| boolean | [**hasController**](http://docs.google.com/javax/imageio/IIOParam.html#hasController())()            Returns true if there is a controller installed for this IIOParam object. |
| void | [**setController**](http://docs.google.com/javax/imageio/IIOParam.html#setController(javax.imageio.IIOParamController))([IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) controller)            Sets the IIOParamController to be used to provide settings for this IIOParam object when the activateController method is called, overriding any default controller. |
| void | [**setDestinationOffset**](http://docs.google.com/javax/imageio/IIOParam.html#setDestinationOffset(java.awt.Point))([Point](http://docs.google.com/java/awt/Point.html) destinationOffset)            Specifies the offset in the destination image at which future decoded pixels are to be placed, when reading, or where a region will be written, when writing. |
| void | [**setDestinationType**](http://docs.google.com/javax/imageio/IIOParam.html#setDestinationType(javax.imageio.ImageTypeSpecifier))([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) destinationType)            Sets the desired image type for the destination image, using an ImageTypeSpecifier. |
| void | [**setSourceBands**](http://docs.google.com/javax/imageio/IIOParam.html#setSourceBands(int%5B%5D))(int[] sourceBands)            Sets the indices of the source bands to be used. |
| void | [**setSourceRegion**](http://docs.google.com/javax/imageio/IIOParam.html#setSourceRegion(java.awt.Rectangle))([Rectangle](http://docs.google.com/java/awt/Rectangle.html) sourceRegion)            Sets the source region of interest. |
| void | [**setSourceSubsampling**](http://docs.google.com/javax/imageio/IIOParam.html#setSourceSubsampling(int,%20int,%20int,%20int))(int sourceXSubsampling, int sourceYSubsampling, int subsamplingXOffset, int subsamplingYOffset)            Specifies a decimation subsampling to apply on I/O. |

| **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** |
| --- |

### sourceRegion

protected [Rectangle](http://docs.google.com/java/awt/Rectangle.html) **sourceRegion**

The source region, on null if none is set.

### sourceXSubsampling

protected int **sourceXSubsampling**

The decimation subsampling to be applied in the horizontal direction. By default, the value is 1. The value must not be negative or 0.

### sourceYSubsampling

protected int **sourceYSubsampling**

The decimation subsampling to be applied in the vertical direction. By default, the value is 1. The value must not be negative or 0.

### subsamplingXOffset

protected int **subsamplingXOffset**

A horizontal offset to be applied to the subsampling grid before subsampling. The first pixel to be used will be offset this amount from the origin of the region, or of the image if no region is specified.

### subsamplingYOffset

protected int **subsamplingYOffset**

A vertical offset to be applied to the subsampling grid before subsampling. The first pixel to be used will be offset this amount from the origin of the region, or of the image if no region is specified.

### sourceBands

protected int[] **sourceBands**

An array of ints indicating which source bands will be used, or null. If null, the set of source bands to be used is as described in the comment for the setSourceBands method. No value should be allowed to be negative.

### destinationType

protected [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) **destinationType**

An ImageTypeSpecifier to be used to generate a destination image when reading, or to set the output color type when writing. If non has been setm the value will be null. By default, the value is null.

### destinationOffset

protected [Point](http://docs.google.com/java/awt/Point.html) **destinationOffset**

The offset in the destination where the upper-left decoded pixel should be placed. By default, the value is (0, 0).

### defaultController

protected [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) **defaultController**

The default IIOParamController that will be used to provide settings for this IIOParam object when the activateController method is called. This default should be set by subclasses that choose to provide their own default controller, usually a GUI, for setting parameters.

**See Also:**[IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html), [getDefaultController()](http://docs.google.com/javax/imageio/IIOParam.html#getDefaultController()), [activateController()](http://docs.google.com/javax/imageio/IIOParam.html#activateController())

### controller

protected [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) **controller**

The IIOParamController that will be used to provide settings for this IIOParam object when the activateController method is called. This value overrides any default controller, even when null.

**See Also:**[IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html), [setController(IIOParamController)](http://docs.google.com/javax/imageio/IIOParam.html#setController(javax.imageio.IIOParamController)), [hasController()](http://docs.google.com/javax/imageio/IIOParam.html#hasController()), [activateController()](http://docs.google.com/javax/imageio/IIOParam.html#activateController())

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

### IIOParam

protected **IIOParam**()

Protected constructor may be called only by subclasses.

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

### setSourceRegion

public void **setSourceRegion**([Rectangle](http://docs.google.com/java/awt/Rectangle.html) sourceRegion)

Sets the source region of interest. The region of interest is described as a rectangle, with the upper-left corner of the source image as pixel (0, 0) and increasing values down and to the right. The actual number of pixels used will depend on the subsampling factors set by setSourceSubsampling. If subsampling has been set such that this number is zero, an IllegalStateException will be thrown.

The source region of interest specified by this method will be clipped as needed to fit within the source bounds, as well as the destination offsets, width, and height at the time of actual I/O.

A value of null for sourceRegion will remove any region specification, causing the entire image to be used.

**Parameters:**sourceRegion - a Rectangle specifying the source region of interest, or null. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if sourceRegion is non-null and either sourceRegion.x or sourceRegion.y is negative. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if sourceRegion is non-null and either sourceRegion.width or sourceRegion.height is negative or 0. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if subsampling is such that this region will have a subsampled width or height of zero.**See Also:**[getSourceRegion()](http://docs.google.com/javax/imageio/IIOParam.html#getSourceRegion()), [setSourceSubsampling(int, int, int, int)](http://docs.google.com/javax/imageio/IIOParam.html#setSourceSubsampling(int,%20int,%20int,%20int)), [setDestinationOffset(java.awt.Point)](http://docs.google.com/javax/imageio/IIOParam.html#setDestinationOffset(java.awt.Point)), [getDestinationOffset()](http://docs.google.com/javax/imageio/IIOParam.html#getDestinationOffset())

### getSourceRegion

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

Returns the source region to be used. The returned value is that set by the most recent call to setSourceRegion, and will be null if there is no region set.

**Returns:**the source region of interest as a Rectangle, or null.**See Also:**[setSourceRegion(java.awt.Rectangle)](http://docs.google.com/javax/imageio/IIOParam.html#setSourceRegion(java.awt.Rectangle))

### setSourceSubsampling

public void **setSourceSubsampling**(int sourceXSubsampling,  
 int sourceYSubsampling,  
 int subsamplingXOffset,  
 int subsamplingYOffset)

Specifies a decimation subsampling to apply on I/O. The sourceXSubsampling and sourceYSubsampling parameters specify the subsampling period (*i.e.*, the number of rows and columns to advance after every source pixel). Specifically, a period of 1 will use every row or column; a period of 2 will use every other row or column. The subsamplingXOffset and subsamplingYOffset parameters specify an offset from the region (or image) origin for the first subsampled pixel. Adjusting the origin of the subsample grid is useful for avoiding seams when subsampling a very large source image into destination regions that will be assembled into a complete subsampled image. Most users will want to simply leave these parameters at 0.

The number of pixels and scanlines to be used are calculated as follows.

The number of subsampled pixels in a scanline is given by

truncate[(width - subsamplingXOffset + sourceXSubsampling - 1) / sourceXSubsampling].

If the region is such that this width is zero, an IllegalStateException is thrown.

The number of scanlines to be used can be computed similarly.

The ability to set the subsampling grid to start somewhere other than the source region origin is useful if the region is being used to create subsampled tiles of a large image, where the tile width and height are not multiples of the subsampling periods. If the subsampling grid does not remain consistent from tile to tile, there will be artifacts at the tile boundaries. By adjusting the subsampling grid offset for each tile to compensate, these artifacts can be avoided. The tradeoff is that in order to avoid these artifacts, the tiles are not all the same size. The grid offset to use in this case is given by:

grid offset = [period - (region offset modulo period)] modulo period)

If either sourceXSubsampling or sourceYSubsampling is 0 or negative, an IllegalArgumentException will be thrown.

If either subsamplingXOffset or subsamplingYOffset is negative or greater than or equal to the corresponding period, an IllegalArgumentException will be thrown.

There is no unsetSourceSubsampling method; simply call setSourceSubsampling(1, 1, 0, 0) to restore default values.

**Parameters:**sourceXSubsampling - the number of columns to advance between pixels.sourceYSubsampling - the number of rows to advance between pixels.subsamplingXOffset - the horizontal offset of the first subsample within the region, or within the image if no region is set.subsamplingYOffset - the horizontal offset of the first subsample within the region, or within the image if no region is set. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if either period is negative or 0, or if either grid offset is negative or greater than the corresponding period. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the source region is such that the subsampled output would contain no pixels.

### getSourceXSubsampling

public int **getSourceXSubsampling**()

Returns the number of source columns to advance for each pixel.

If setSourceSubsampling has not been called, 1 is returned (which is the correct value).

**Returns:**the source subsampling X period.**See Also:**[setSourceSubsampling(int, int, int, int)](http://docs.google.com/javax/imageio/IIOParam.html#setSourceSubsampling(int,%20int,%20int,%20int)), [getSourceYSubsampling()](http://docs.google.com/javax/imageio/IIOParam.html#getSourceYSubsampling())

### getSourceYSubsampling

public int **getSourceYSubsampling**()

Returns the number of rows to advance for each pixel.

If setSourceSubsampling has not been called, 1 is returned (which is the correct value).

**Returns:**the source subsampling Y period.**See Also:**[setSourceSubsampling(int, int, int, int)](http://docs.google.com/javax/imageio/IIOParam.html#setSourceSubsampling(int,%20int,%20int,%20int)), [getSourceXSubsampling()](http://docs.google.com/javax/imageio/IIOParam.html#getSourceXSubsampling())

### getSubsamplingXOffset

public int **getSubsamplingXOffset**()

Returns the horizontal offset of the subsampling grid.

If setSourceSubsampling has not been called, 0 is returned (which is the correct value).

**Returns:**the source subsampling grid X offset.**See Also:**[setSourceSubsampling(int, int, int, int)](http://docs.google.com/javax/imageio/IIOParam.html#setSourceSubsampling(int,%20int,%20int,%20int)), [getSubsamplingYOffset()](http://docs.google.com/javax/imageio/IIOParam.html#getSubsamplingYOffset())

### getSubsamplingYOffset

public int **getSubsamplingYOffset**()

Returns the vertical offset of the subsampling grid.

If setSourceSubsampling has not been called, 0 is returned (which is the correct value).

**Returns:**the source subsampling grid Y offset.**See Also:**[setSourceSubsampling(int, int, int, int)](http://docs.google.com/javax/imageio/IIOParam.html#setSourceSubsampling(int,%20int,%20int,%20int)), [getSubsamplingXOffset()](http://docs.google.com/javax/imageio/IIOParam.html#getSubsamplingXOffset())

### setSourceBands

public void **setSourceBands**(int[] sourceBands)

Sets the indices of the source bands to be used. Duplicate indices are not allowed.

A null value indicates that all source bands will be used.

At the time of reading, an IllegalArgumentException will be thrown by the reader or writer if a value larger than the largest available source band index has been specified or if the number of source bands and destination bands to be used differ. The ImageReader.checkReadParamBandSettings method may be used to automate this test.

Semantically, a copy is made of the array; changes to the array contents subsequent to this call have no effect on this IIOParam.

**Parameters:**sourceBands - an array of integer band indices to be used. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if sourceBands contains a negative or duplicate value.**See Also:**[getSourceBands()](http://docs.google.com/javax/imageio/IIOParam.html#getSourceBands()), [ImageReadParam.setDestinationBands(int[])](http://docs.google.com/javax/imageio/ImageReadParam.html#setDestinationBands(int%5B%5D)), [ImageReader.checkReadParamBandSettings(javax.imageio.ImageReadParam, int, int)](http://docs.google.com/javax/imageio/ImageReader.html#checkReadParamBandSettings(javax.imageio.ImageReadParam,%20int,%20int))

### getSourceBands

public int[] **getSourceBands**()

Returns the set of of source bands to be used. The returned value is that set by the most recent call to setSourceBands, or null if there have been no calls to setSourceBands.

Semantically, the array returned is a copy; changes to array contents subsequent to this call have no effect on this IIOParam.

**Returns:**the set of source bands to be used, or null.**See Also:**[setSourceBands(int[])](http://docs.google.com/javax/imageio/IIOParam.html#setSourceBands(int%5B%5D))

### setDestinationType

public void **setDestinationType**([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) destinationType)

Sets the desired image type for the destination image, using an ImageTypeSpecifier.

When reading, if the layout of the destination has been set using this method, each call to an ImageReader read method will return a new BufferedImage using the format specified by the supplied type specifier. As a side effect, any destination BufferedImage set by ImageReadParam.setDestination(BufferedImage) will no longer be set as the destination. In other words, this method may be thought of as calling setDestination((BufferedImage)null).

When writing, the destination type maybe used to determine the color type of the image. The SampleModel information will be ignored, and may be null. For example, a 4-banded image could represent either CMYK or RGBA data. If a destination type is set, its ColorModel will override any ColorModel on the image itself. This is crucial when setSourceBands is used since the image's ColorModel will refer to the entire image rather than to the subset of bands being written.

**Parameters:**destinationType - the ImageTypeSpecifier to be used to determine the destination layout and color type.**See Also:**[getDestinationType()](http://docs.google.com/javax/imageio/IIOParam.html#getDestinationType())

### getDestinationType

public [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) **getDestinationType**()

Returns the type of image to be returned by the read, if one was set by a call to setDestination(ImageTypeSpecifier), as an ImageTypeSpecifier. If none was set, null is returned.

**Returns:**an ImageTypeSpecifier describing the destination type, or null.**See Also:**[setDestinationType(javax.imageio.ImageTypeSpecifier)](http://docs.google.com/javax/imageio/IIOParam.html#setDestinationType(javax.imageio.ImageTypeSpecifier))

### setDestinationOffset

public void **setDestinationOffset**([Point](http://docs.google.com/java/awt/Point.html) destinationOffset)

Specifies the offset in the destination image at which future decoded pixels are to be placed, when reading, or where a region will be written, when writing.

When reading, the region to be written within the destination BufferedImage will start at this offset and have a width and height determined by the source region of interest, the subsampling parameters, and the destination bounds.

Normal writes are not affected by this method, only writes performed using ImageWriter.replacePixels. For such writes, the offset specified is within the output stream image whose pixels are being modified.

There is no unsetDestinationOffset method; simply call setDestinationOffset(new Point(0, 0)) to restore default values.

**Parameters:**destinationOffset - the offset in the destination, as a Point. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if destinationOffset is null.**See Also:**[getDestinationOffset()](http://docs.google.com/javax/imageio/IIOParam.html#getDestinationOffset()), [ImageWriter.replacePixels(java.awt.image.RenderedImage, javax.imageio.ImageWriteParam)](http://docs.google.com/javax/imageio/ImageWriter.html#replacePixels(java.awt.image.RenderedImage,%20javax.imageio.ImageWriteParam))

### getDestinationOffset

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

Returns the offset in the destination image at which pixels are to be placed.

If setDestinationOffsets has not been called, a Point with zero X and Y values is returned (which is the correct value).

**Returns:**the destination offset as a Point.**See Also:**[setDestinationOffset(java.awt.Point)](http://docs.google.com/javax/imageio/IIOParam.html#setDestinationOffset(java.awt.Point))

### setController

public void **setController**([IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) controller)

Sets the IIOParamController to be used to provide settings for this IIOParam object when the activateController method is called, overriding any default controller. If the argument is null, no controller will be used, including any default. To restore the default, use setController(getDefaultController()).

**Parameters:**controller - An appropriate IIOParamController, or null.**See Also:**[IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html), [getController()](http://docs.google.com/javax/imageio/IIOParam.html#getController()), [getDefaultController()](http://docs.google.com/javax/imageio/IIOParam.html#getDefaultController()), [hasController()](http://docs.google.com/javax/imageio/IIOParam.html#hasController()), [activateController()](http://docs.google.com/javax/imageio/IIOParam.html#activateController())

### getController

public [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) **getController**()

Returns whatever IIOParamController is currently installed. This could be the default if there is one, null, or the argument of the most recent call to setController.

**Returns:**the currently installed IIOParamController, or null.**See Also:**[IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html), [setController(javax.imageio.IIOParamController)](http://docs.google.com/javax/imageio/IIOParam.html#setController(javax.imageio.IIOParamController)), [getDefaultController()](http://docs.google.com/javax/imageio/IIOParam.html#getDefaultController()), [hasController()](http://docs.google.com/javax/imageio/IIOParam.html#hasController()), [activateController()](http://docs.google.com/javax/imageio/IIOParam.html#activateController())

### getDefaultController

public [IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html) **getDefaultController**()

Returns the default IIOParamController, if there is one, regardless of the currently installed controller. If there is no default controller, returns null.

**Returns:**the default IIOParamController, or null.**See Also:**[IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html), [setController(IIOParamController)](http://docs.google.com/javax/imageio/IIOParam.html#setController(javax.imageio.IIOParamController)), [getController()](http://docs.google.com/javax/imageio/IIOParam.html#getController()), [hasController()](http://docs.google.com/javax/imageio/IIOParam.html#hasController()), [activateController()](http://docs.google.com/javax/imageio/IIOParam.html#activateController())

### hasController

public boolean **hasController**()

Returns true if there is a controller installed for this IIOParam object. This will return true if getController would not return null.

**Returns:**true if a controller is installed.**See Also:**[IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html), [setController(IIOParamController)](http://docs.google.com/javax/imageio/IIOParam.html#setController(javax.imageio.IIOParamController)), [getController()](http://docs.google.com/javax/imageio/IIOParam.html#getController()), [getDefaultController()](http://docs.google.com/javax/imageio/IIOParam.html#getDefaultController()), [activateController()](http://docs.google.com/javax/imageio/IIOParam.html#activateController())

### activateController

public boolean **activateController**()

Activates the installed IIOParamController for this IIOParam object and returns the resulting value. When this method returns true, all values for this IIOParam object will be ready for the next read or write operation. If false is returned, no settings in this object will have been disturbed (*i.e.*, the user canceled the operation).

Ordinarily, the controller will be a GUI providing a user interface for a subclass of IIOParam for a particular plug-in. Controllers need not be GUIs, however.

**Returns:**true if the controller completed normally. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if there is no controller currently installed.**See Also:**[IIOParamController](http://docs.google.com/javax/imageio/IIOParamController.html), [setController(IIOParamController)](http://docs.google.com/javax/imageio/IIOParam.html#setController(javax.imageio.IIOParamController)), [getController()](http://docs.google.com/javax/imageio/IIOParam.html#getController()), [getDefaultController()](http://docs.google.com/javax/imageio/IIOParam.html#getDefaultController()), [hasController()](http://docs.google.com/javax/imageio/IIOParam.html#hasController())

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/IIOParam.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/imageio/IIOImage.html)   [**NEXT CLASS**](http://docs.google.com/javax/imageio/IIOParamController.html) | [**FRAMES**](http://docs.google.com/index.html?javax/imageio/IIOParam.html)    [**NO FRAMES**](http://docs.google.com/IIOParam.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#z337ya) | [METHOD](#1y810tw) |

[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).