| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ImageWriter.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/ImageWriteParam.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/imageio/ImageWriter.html)    [**NO FRAMES**](http://docs.google.com/ImageWriter.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#1ksv4uv) | [METHOD](#2jxsxqh) |

## **javax.imageio**

Class ImageWriter

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

**All Implemented Interfaces:** [ImageTranscoder](http://docs.google.com/javax/imageio/ImageTranscoder.html)

public abstract class **ImageWriter**extends [Object](http://docs.google.com/java/lang/Object.html)implements [ImageTranscoder](http://docs.google.com/javax/imageio/ImageTranscoder.html)

An abstract superclass for encoding and writing images. This class must be subclassed by classes that write out images in the context of the Java Image I/O framework.

ImageWriter objects are normally instantiated by the service provider class for the specific format. Service provider classes are registered with the IIORegistry, which uses them for format recognition and presentation of available format readers and writers.

**See Also:**[ImageReader](http://docs.google.com/javax/imageio/ImageReader.html), [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html), [IIORegistry](http://docs.google.com/javax/imageio/spi/IIORegistry.html), [ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html)

| **Field Summary** | |
| --- | --- |
| protected  [Locale](http://docs.google.com/java/util/Locale.html)[] | [**availableLocales**](http://docs.google.com/javax/imageio/ImageWriter.html#availableLocales)            An array of Locales that may be used to localize warning messages and compression setting values, or null if localization is not supported. |
| protected  [Locale](http://docs.google.com/java/util/Locale.html) | [**locale**](http://docs.google.com/javax/imageio/ImageWriter.html#locale)            The current Locale to be used for localization, or null if none has been set. |
| protected  [ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html) | [**originatingProvider**](http://docs.google.com/javax/imageio/ImageWriter.html#originatingProvider)            The ImageWriterSpi that instantiated this object, or null if its identity is not known or none exists. |
| protected  [Object](http://docs.google.com/java/lang/Object.html) | [**output**](http://docs.google.com/javax/imageio/ImageWriter.html#output)            The ImageOutputStream or other Object set by setOutput and retrieved by getOutput. |
| protected  [List](http://docs.google.com/java/util/List.html)<[IIOWriteProgressListener](http://docs.google.com/javax/imageio/event/IIOWriteProgressListener.html)> | [**progressListeners**](http://docs.google.com/javax/imageio/ImageWriter.html#progressListeners)            A List of currently registered IIOWriteProgressListeners, initialized by default null, which is synonymous with an empty List. |
| protected  [List](http://docs.google.com/java/util/List.html)<[IIOWriteWarningListener](http://docs.google.com/javax/imageio/event/IIOWriteWarningListener.html)> | [**warningListeners**](http://docs.google.com/javax/imageio/ImageWriter.html#warningListeners)            A List of currently registered IIOWriteWarningListeners, initialized by default to null, which is synonymous with an empty List. |
| protected  [List](http://docs.google.com/java/util/List.html)<[Locale](http://docs.google.com/java/util/Locale.html)> | [**warningLocales**](http://docs.google.com/javax/imageio/ImageWriter.html#warningLocales)            A List of Locales, one for each element of warningListeners, initialized by default null, which is synonymous with an empty List. |

| **Constructor Summary** | |
| --- | --- |
| protected | [**ImageWriter**](http://docs.google.com/javax/imageio/ImageWriter.html#ImageWriter(javax.imageio.spi.ImageWriterSpi))([ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html) originatingProvider)            Constructs an ImageWriter and sets its originatingProvider instance variable to the supplied value. |

| **Method Summary** | |
| --- | --- |
| void | [**abort**](http://docs.google.com/javax/imageio/ImageWriter.html#abort())()            Requests that any current write operation be aborted. |
| protected  boolean | [**abortRequested**](http://docs.google.com/javax/imageio/ImageWriter.html#abortRequested())()            Returns true if a request to abort the current write operation has been made since the writer was instantiated or clearAbortRequest was called. |
| void | [**addIIOWriteProgressListener**](http://docs.google.com/javax/imageio/ImageWriter.html#addIIOWriteProgressListener(javax.imageio.event.IIOWriteProgressListener))([IIOWriteProgressListener](http://docs.google.com/javax/imageio/event/IIOWriteProgressListener.html) listener)            Adds an IIOWriteProgressListener to the list of registered progress listeners. |
| void | [**addIIOWriteWarningListener**](http://docs.google.com/javax/imageio/ImageWriter.html#addIIOWriteWarningListener(javax.imageio.event.IIOWriteWarningListener))([IIOWriteWarningListener](http://docs.google.com/javax/imageio/event/IIOWriteWarningListener.html) listener)            Adds an IIOWriteWarningListener to the list of registered warning listeners. |
| boolean | [**canInsertEmpty**](http://docs.google.com/javax/imageio/ImageWriter.html#canInsertEmpty(int))(int imageIndex)            Returns true if the writer supports the insertion of a new, empty image at the given index. |
| boolean | [**canInsertImage**](http://docs.google.com/javax/imageio/ImageWriter.html#canInsertImage(int))(int imageIndex)            Returns true if the writer supports the insertion of a new image at the given index. |
| boolean | [**canRemoveImage**](http://docs.google.com/javax/imageio/ImageWriter.html#canRemoveImage(int))(int imageIndex)            Returns true if the writer supports the removal of an existing image at the given index. |
| boolean | [**canReplaceImageMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#canReplaceImageMetadata(int))(int imageIndex)            Returns true if it is possible to replace the image metadata associated with an existing image with index imageIndex. |
| boolean | [**canReplacePixels**](http://docs.google.com/javax/imageio/ImageWriter.html#canReplacePixels(int))(int imageIndex)            Returns true if the writer allows pixels of the given image to be replaced using the replacePixels methods. |
| boolean | [**canReplaceStreamMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#canReplaceStreamMetadata())()            Returns true if it is possible to replace the stream metadata already present in the output. |
| boolean | [**canWriteEmpty**](http://docs.google.com/javax/imageio/ImageWriter.html#canWriteEmpty())()            Returns true if the writer supports the writing of a complete image stream consisting of a single image with undefined pixel values and associated metadata and thumbnails to the output. |
| boolean | [**canWriteRasters**](http://docs.google.com/javax/imageio/ImageWriter.html#canWriteRasters())()            Returns true if the methods that take an IIOImage parameter are capable of dealing with a Raster (as opposed to RenderedImage) source image. |
| boolean | [**canWriteSequence**](http://docs.google.com/javax/imageio/ImageWriter.html#canWriteSequence())()            Returns true if the writer is able to append an image to an image stream that already contains header information and possibly prior images. |
| protected  void | [**clearAbortRequest**](http://docs.google.com/javax/imageio/ImageWriter.html#clearAbortRequest())()            Clears any previous abort request. |
| abstract  [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) | [**convertImageMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#convertImageMetadata(javax.imageio.metadata.IIOMetadata,%20javax.imageio.ImageTypeSpecifier,%20javax.imageio.ImageWriteParam))([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) inData, [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Returns an IIOMetadata object that may be used for encoding and optionally modified using its document interfaces or other interfaces specific to the writer plug-in that will be used for encoding. |
| abstract  [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) | [**convertStreamMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#convertStreamMetadata(javax.imageio.metadata.IIOMetadata,%20javax.imageio.ImageWriteParam))([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) inData, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Returns an IIOMetadata object that may be used for encoding and optionally modified using its document interfaces or other interfaces specific to the writer plug-in that will be used for encoding. |
| void | [**dispose**](http://docs.google.com/javax/imageio/ImageWriter.html#dispose())()            Allows any resources held by this object to be released. |
| void | [**endInsertEmpty**](http://docs.google.com/javax/imageio/ImageWriter.html#endInsertEmpty())()            Completes the insertion of a new image that was begun with a prior call to prepareInsertEmpty. |
| void | [**endReplacePixels**](http://docs.google.com/javax/imageio/ImageWriter.html#endReplacePixels())()            Terminates a sequence of calls to replacePixels. |
| void | [**endWriteEmpty**](http://docs.google.com/javax/imageio/ImageWriter.html#endWriteEmpty())()            Completes the writing of a new image that was begun with a prior call to prepareWriteEmpty. |
| void | [**endWriteSequence**](http://docs.google.com/javax/imageio/ImageWriter.html#endWriteSequence())()            Completes the writing of a sequence of images begun with prepareWriteSequence. |
| [Locale](http://docs.google.com/java/util/Locale.html)[] | [**getAvailableLocales**](http://docs.google.com/javax/imageio/ImageWriter.html#getAvailableLocales())()            Returns an array of Locales that may be used to localize warning listeners and compression settings. |
| abstract  [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) | [**getDefaultImageMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#getDefaultImageMetadata(javax.imageio.ImageTypeSpecifier,%20javax.imageio.ImageWriteParam))([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Returns an IIOMetadata object containing default values for encoding an image of the given type. |
| abstract  [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) | [**getDefaultStreamMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#getDefaultStreamMetadata(javax.imageio.ImageWriteParam))([ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Returns an IIOMetadata object containing default values for encoding a stream of images. |
| [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) | [**getDefaultWriteParam**](http://docs.google.com/javax/imageio/ImageWriter.html#getDefaultWriteParam())()            Returns a new ImageWriteParam object of the appropriate type for this file format containing default values, that is, those values that would be used if no ImageWriteParam object were specified. |
| [Locale](http://docs.google.com/java/util/Locale.html) | [**getLocale**](http://docs.google.com/javax/imageio/ImageWriter.html#getLocale())()            Returns the currently set Locale, or null if none has been set. |
| int | [**getNumThumbnailsSupported**](http://docs.google.com/javax/imageio/ImageWriter.html#getNumThumbnailsSupported(javax.imageio.ImageTypeSpecifier,%20javax.imageio.ImageWriteParam,%20javax.imageio.metadata.IIOMetadata,%20javax.imageio.metadata.IIOMetadata))([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param, [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata, [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata)            Returns the number of thumbnails suported by the format being written, given the image type and any additional write parameters and metadata objects that will be used during encoding. |
| [ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html) | [**getOriginatingProvider**](http://docs.google.com/javax/imageio/ImageWriter.html#getOriginatingProvider())()            Returns the ImageWriterSpi object that created this ImageWriter, or null if this object was not created through the IIORegistry. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getOutput**](http://docs.google.com/javax/imageio/ImageWriter.html#getOutput())()            Returns the ImageOutputStream or other Object set by the most recent call to the setOutput method. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html)[] | [**getPreferredThumbnailSizes**](http://docs.google.com/javax/imageio/ImageWriter.html#getPreferredThumbnailSizes(javax.imageio.ImageTypeSpecifier,%20javax.imageio.ImageWriteParam,%20javax.imageio.metadata.IIOMetadata,%20javax.imageio.metadata.IIOMetadata))([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param, [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata, [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata)            Returns an array of Dimensions indicating the legal size ranges for thumbnail images as they will be encoded in the output file or stream. |
| void | [**prepareInsertEmpty**](http://docs.google.com/javax/imageio/ImageWriter.html#prepareInsertEmpty(int,%20javax.imageio.ImageTypeSpecifier,%20int,%20int,%20javax.imageio.metadata.IIOMetadata,%20java.util.List,%20javax.imageio.ImageWriteParam))(int imageIndex, [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType, int width, int height, [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata, [List](http://docs.google.com/java/util/List.html)<? extends [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html)> thumbnails, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Begins the insertion of a new image with undefined pixel values into an existing image stream. |
| void | [**prepareReplacePixels**](http://docs.google.com/javax/imageio/ImageWriter.html#prepareReplacePixels(int,%20java.awt.Rectangle))(int imageIndex, [Rectangle](http://docs.google.com/java/awt/Rectangle.html) region)            Prepares the writer to handle a series of calls to the replacePixels methods. |
| void | [**prepareWriteEmpty**](http://docs.google.com/javax/imageio/ImageWriter.html#prepareWriteEmpty(javax.imageio.metadata.IIOMetadata,%20javax.imageio.ImageTypeSpecifier,%20int,%20int,%20javax.imageio.metadata.IIOMetadata,%20java.util.List,%20javax.imageio.ImageWriteParam))([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata, [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType, int width, int height, [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata, [List](http://docs.google.com/java/util/List.html)<? extends [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html)> thumbnails, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Begins the writing of a complete image stream, consisting of a single image with undefined pixel values and associated metadata and thumbnails, to the output. |
| void | [**prepareWriteSequence**](http://docs.google.com/javax/imageio/ImageWriter.html#prepareWriteSequence(javax.imageio.metadata.IIOMetadata))([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata)            Prepares a stream to accept a series of subsequent writeToSequence calls, using the provided stream metadata object. |
| protected  void | [**processImageComplete**](http://docs.google.com/javax/imageio/ImageWriter.html#processImageComplete())()            Broadcasts the completion of an image write to all registered IIOWriteProgressListeners by calling their imageComplete method. |
| protected  void | [**processImageProgress**](http://docs.google.com/javax/imageio/ImageWriter.html#processImageProgress(float))(float percentageDone)            Broadcasts the current percentage of image completion to all registered IIOWriteProgressListeners by calling their imageProgress method. |
| protected  void | [**processImageStarted**](http://docs.google.com/javax/imageio/ImageWriter.html#processImageStarted(int))(int imageIndex)            Broadcasts the start of an image write to all registered IIOWriteProgressListeners by calling their imageStarted method. |
| protected  void | [**processThumbnailComplete**](http://docs.google.com/javax/imageio/ImageWriter.html#processThumbnailComplete())()            Broadcasts the completion of a thumbnail write to all registered IIOWriteProgressListeners by calling their thumbnailComplete method. |
| protected  void | [**processThumbnailProgress**](http://docs.google.com/javax/imageio/ImageWriter.html#processThumbnailProgress(float))(float percentageDone)            Broadcasts the current percentage of thumbnail completion to all registered IIOWriteProgressListeners by calling their thumbnailProgress method. |
| protected  void | [**processThumbnailStarted**](http://docs.google.com/javax/imageio/ImageWriter.html#processThumbnailStarted(int,%20int))(int imageIndex, int thumbnailIndex)            Broadcasts the start of a thumbnail write to all registered IIOWriteProgressListeners by calling their thumbnailStarted method. |
| protected  void | [**processWarningOccurred**](http://docs.google.com/javax/imageio/ImageWriter.html#processWarningOccurred(int,%20java.lang.String))(int imageIndex, [String](http://docs.google.com/java/lang/String.html) warning)            Broadcasts a warning message to all registered IIOWriteWarningListeners by calling their warningOccurred method. |
| protected  void | [**processWarningOccurred**](http://docs.google.com/javax/imageio/ImageWriter.html#processWarningOccurred(int,%20java.lang.String,%20java.lang.String))(int imageIndex, [String](http://docs.google.com/java/lang/String.html) baseName, [String](http://docs.google.com/java/lang/String.html) keyword)            Broadcasts a localized warning message to all registered IIOWriteWarningListeners by calling their warningOccurred method with a string taken from a ResourceBundle. |
| protected  void | [**processWriteAborted**](http://docs.google.com/javax/imageio/ImageWriter.html#processWriteAborted())()            Broadcasts that the write has been aborted to all registered IIOWriteProgressListeners by calling their writeAborted method. |
| void | [**removeAllIIOWriteProgressListeners**](http://docs.google.com/javax/imageio/ImageWriter.html#removeAllIIOWriteProgressListeners())()            Removes all currently registered IIOWriteProgressListener objects. |
| void | [**removeAllIIOWriteWarningListeners**](http://docs.google.com/javax/imageio/ImageWriter.html#removeAllIIOWriteWarningListeners())()            Removes all currently registered IIOWriteWarningListener objects. |
| void | [**removeIIOWriteProgressListener**](http://docs.google.com/javax/imageio/ImageWriter.html#removeIIOWriteProgressListener(javax.imageio.event.IIOWriteProgressListener))([IIOWriteProgressListener](http://docs.google.com/javax/imageio/event/IIOWriteProgressListener.html) listener)            Removes an IIOWriteProgressListener from the list of registered progress listeners. |
| void | [**removeIIOWriteWarningListener**](http://docs.google.com/javax/imageio/ImageWriter.html#removeIIOWriteWarningListener(javax.imageio.event.IIOWriteWarningListener))([IIOWriteWarningListener](http://docs.google.com/javax/imageio/event/IIOWriteWarningListener.html) listener)            Removes an IIOWriteWarningListener from the list of registered warning listeners. |
| void | [**removeImage**](http://docs.google.com/javax/imageio/ImageWriter.html#removeImage(int))(int imageIndex)            Removes an image from the stream. |
| void | [**replaceImageMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#replaceImageMetadata(int,%20javax.imageio.metadata.IIOMetadata))(int imageIndex, [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata)            Replaces the image metadata associated with an existing image. |
| void | [**replacePixels**](http://docs.google.com/javax/imageio/ImageWriter.html#replacePixels(java.awt.image.Raster,%20javax.imageio.ImageWriteParam))([Raster](http://docs.google.com/java/awt/image/Raster.html) raster, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Replaces a portion of an image already present in the output with a portion of the given Raster. |
| void | [**replacePixels**](http://docs.google.com/javax/imageio/ImageWriter.html#replacePixels(java.awt.image.RenderedImage,%20javax.imageio.ImageWriteParam))([RenderedImage](http://docs.google.com/java/awt/image/RenderedImage.html) image, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Replaces a portion of an image already present in the output with a portion of the given image. |
| void | [**replaceStreamMetadata**](http://docs.google.com/javax/imageio/ImageWriter.html#replaceStreamMetadata(javax.imageio.metadata.IIOMetadata))([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata)            Replaces the stream metadata in the output with new information. |
| void | [**reset**](http://docs.google.com/javax/imageio/ImageWriter.html#reset())()            Restores the ImageWriter to its initial state. |
| void | [**setLocale**](http://docs.google.com/javax/imageio/ImageWriter.html#setLocale(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) locale)            Sets the current Locale of this ImageWriter to the given value. |
| void | [**setOutput**](http://docs.google.com/javax/imageio/ImageWriter.html#setOutput(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) output)            Sets the destination to the given ImageOutputStream or other Object. |
| void | [**write**](http://docs.google.com/javax/imageio/ImageWriter.html#write(javax.imageio.IIOImage))([IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image)            Appends a complete image stream containing a single image with default metadata and thumbnails to the output. |
| abstract  void | [**write**](http://docs.google.com/javax/imageio/ImageWriter.html#write(javax.imageio.metadata.IIOMetadata,%20javax.imageio.IIOImage,%20javax.imageio.ImageWriteParam))([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata, [IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Appends a complete image stream containing a single image and associated stream and image metadata and thumbnails to the output. |
| void | [**write**](http://docs.google.com/javax/imageio/ImageWriter.html#write(java.awt.image.RenderedImage))([RenderedImage](http://docs.google.com/java/awt/image/RenderedImage.html) image)            Appends a complete image stream consisting of a single image with default metadata and thumbnails to the output. |
| void | [**writeInsert**](http://docs.google.com/javax/imageio/ImageWriter.html#writeInsert(int,%20javax.imageio.IIOImage,%20javax.imageio.ImageWriteParam))(int imageIndex, [IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Inserts a new image into an existing image stream. |
| void | [**writeToSequence**](http://docs.google.com/javax/imageio/ImageWriter.html#writeToSequence(javax.imageio.IIOImage,%20javax.imageio.ImageWriteParam))([IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image, [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)            Appends a single image and possibly associated metadata and thumbnails, to the output. |

| **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** |
| --- |

### originatingProvider

protected [ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html) **originatingProvider**

The ImageWriterSpi that instantiated this object, or null if its identity is not known or none exists. By default it is initialized to null.

### output

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

The ImageOutputStream or other Object set by setOutput and retrieved by getOutput. By default it is initialized to null.

### availableLocales

protected [Locale](http://docs.google.com/java/util/Locale.html)[] **availableLocales**

An array of Locales that may be used to localize warning messages and compression setting values, or null if localization is not supported. By default it is initialized to null.

### locale

protected [Locale](http://docs.google.com/java/util/Locale.html) **locale**

The current Locale to be used for localization, or null if none has been set. By default it is initialized to null.

### warningListeners

protected [List](http://docs.google.com/java/util/List.html)<[IIOWriteWarningListener](http://docs.google.com/javax/imageio/event/IIOWriteWarningListener.html)> **warningListeners**

A List of currently registered IIOWriteWarningListeners, initialized by default to null, which is synonymous with an empty List.

### warningLocales

protected [List](http://docs.google.com/java/util/List.html)<[Locale](http://docs.google.com/java/util/Locale.html)> **warningLocales**

A List of Locales, one for each element of warningListeners, initialized by default null, which is synonymous with an empty List.

### progressListeners

protected [List](http://docs.google.com/java/util/List.html)<[IIOWriteProgressListener](http://docs.google.com/javax/imageio/event/IIOWriteProgressListener.html)> **progressListeners**

A List of currently registered IIOWriteProgressListeners, initialized by default null, which is synonymous with an empty List.

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

### ImageWriter

protected **ImageWriter**([ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html) originatingProvider)

Constructs an ImageWriter and sets its originatingProvider instance variable to the supplied value.

Subclasses that make use of extensions should provide a constructor with signature (ImageWriterSpi, Object) in order to retrieve the extension object. If the extension object is unsuitable, an IllegalArgumentException should be thrown.

**Parameters:**originatingProvider - the ImageWriterSpi that is constructing this object, or null.

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

### getOriginatingProvider

public [ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html) **getOriginatingProvider**()

Returns the ImageWriterSpi object that created this ImageWriter, or null if this object was not created through the IIORegistry.

The default implementation returns the value of the originatingProvider instance variable.

**Returns:**an ImageWriterSpi, or null.**See Also:**[ImageWriterSpi](http://docs.google.com/javax/imageio/spi/ImageWriterSpi.html)

### setOutput

public void **setOutput**([Object](http://docs.google.com/java/lang/Object.html) output)

Sets the destination to the given ImageOutputStream or other Object. The destination is assumed to be ready to accept data, and will not be closed at the end of each write. This allows distributed imaging applications to transmit a series of images over a single network connection. If output is null, any currently set output will be removed.

If output is an ImageOutputStream, calls to the write, writeToSequence, and prepareWriteEmpty/endWriteEmpty methods will preserve the existing contents of the stream. Other write methods, such as writeInsert, replaceStreamMetadata, replaceImageMetadata, replacePixels, prepareInsertEmpty/endInsertEmpty, and endWriteSequence, require the full contents of the stream to be readable and writable, and may alter any portion of the stream.

Use of a general Object other than an ImageOutputStream is intended for writers that interact directly with an output device or imaging protocol. The set of legal classes is advertised by the writer's service provider's getOutputTypes method; most writers will return a single-element array containing only ImageOutputStream.class to indicate that they accept only an ImageOutputStream.

The default implementation sets the output instance variable to the value of output after checking output against the set of classes advertised by the originating provider, if there is one.

**Parameters:**output - the ImageOutputStream or other Object to use for future writing. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if output is not an instance of one of the classes returned by the originating service provider's getOutputTypes method.**See Also:**[getOutput()](http://docs.google.com/javax/imageio/ImageWriter.html#getOutput())

### getOutput

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

Returns the ImageOutputStream or other Object set by the most recent call to the setOutput method. If no destination has been set, null is returned.

The default implementation returns the value of the output instance variable.

**Returns:**the Object that was specified using setOutput, or null.**See Also:**[setOutput(java.lang.Object)](http://docs.google.com/javax/imageio/ImageWriter.html#setOutput(java.lang.Object))

### getAvailableLocales

public [Locale](http://docs.google.com/java/util/Locale.html)[] **getAvailableLocales**()

Returns an array of Locales that may be used to localize warning listeners and compression settings. A return value of null indicates that localization is not supported.

The default implementation returns a clone of the availableLocales instance variable if it is non-null, or else returns null.

**Returns:**an array of Locales that may be used as arguments to setLocale, or null.

### setLocale

public void **setLocale**([Locale](http://docs.google.com/java/util/Locale.html) locale)

Sets the current Locale of this ImageWriter to the given value. A value of null removes any previous setting, and indicates that the writer should localize as it sees fit.

The default implementation checks locale against the values returned by getAvailableLocales, and sets the locale instance variable if it is found. If locale is null, the instance variable is set to null without any checking.

**Parameters:**locale - the desired Locale, or null. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if locale is non-null but is not one of the values returned by getAvailableLocales.**See Also:**[getLocale()](http://docs.google.com/javax/imageio/ImageWriter.html#getLocale())

### getLocale

public [Locale](http://docs.google.com/java/util/Locale.html) **getLocale**()

Returns the currently set Locale, or null if none has been set.

The default implementation returns the value of the locale instance variable.

**Returns:**the current Locale, or null.**See Also:**[setLocale(java.util.Locale)](http://docs.google.com/javax/imageio/ImageWriter.html#setLocale(java.util.Locale))

### getDefaultWriteParam

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

Returns a new ImageWriteParam object of the appropriate type for this file format containing default values, that is, those values that would be used if no ImageWriteParam object were specified. This is useful as a starting point for tweaking just a few parameters and otherwise leaving the default settings alone.

The default implementation constructs and returns a new ImageWriteParam object that does not allow tiling, progressive encoding, or compression, and that will be localized for the current Locale (*i.e.*, what you would get by calling new ImageWriteParam(getLocale()).

Individual plug-ins may return an instance of ImageWriteParam with additional optional features enabled, or they may return an instance of a plug-in specific subclass of ImageWriteParam.

**Returns:**a new ImageWriteParam object containing default values.

### getDefaultStreamMetadata

public abstract [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) **getDefaultStreamMetadata**([ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)

Returns an IIOMetadata object containing default values for encoding a stream of images. The contents of the object may be manipulated using either the XML tree structure returned by the IIOMetadata.getAsTree method, an IIOMetadataController object, or via plug-in specific interfaces, and the resulting data supplied to one of the write methods that take a stream metadata parameter.

An optional ImageWriteParam may be supplied for cases where it may affect the structure of the stream metadata.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

Writers that do not make use of stream metadata (*e.g.*, writers for single-image formats) should return null.

**Parameters:**param - an ImageWriteParam that will be used to encode the image, or null. **Returns:**an IIOMetadata object.

### getDefaultImageMetadata

public abstract [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) **getDefaultImageMetadata**([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)

Returns an IIOMetadata object containing default values for encoding an image of the given type. The contents of the object may be manipulated using either the XML tree structure returned by the IIOMetadata.getAsTree method, an IIOMetadataController object, or via plug-in specific interfaces, and the resulting data supplied to one of the write methods that take a stream metadata parameter.

An optional ImageWriteParam may be supplied for cases where it may affect the structure of the image metadata.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

**Parameters:**imageType - an ImageTypeSpecifier indicating the format of the image to be written later.param - an ImageWriteParam that will be used to encode the image, or null. **Returns:**an IIOMetadata object.

### convertStreamMetadata

public abstract [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) **convertStreamMetadata**([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) inData,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)

**Description copied from interface:** [**ImageTranscoder**](http://docs.google.com/javax/imageio/ImageTranscoder.html#convertStreamMetadata(javax.imageio.metadata.IIOMetadata,%20javax.imageio.ImageWriteParam)) Returns an IIOMetadata object that may be used for encoding and optionally modified using its document interfaces or other interfaces specific to the writer plug-in that will be used for encoding.

An optional ImageWriteParam may be supplied for cases where it may affect the structure of the stream metadata.

If the supplied ImageWriteParam contains optional setting values not understood by this writer or transcoder, they will be ignored.

**Specified by:**[convertStreamMetadata](http://docs.google.com/javax/imageio/ImageTranscoder.html#convertStreamMetadata(javax.imageio.metadata.IIOMetadata,%20javax.imageio.ImageWriteParam)) in interface [ImageTranscoder](http://docs.google.com/javax/imageio/ImageTranscoder.html) **Parameters:**inData - an IIOMetadata object representing stream metadata, used to initialize the state of the returned object.param - an ImageWriteParam that will be used to encode the image, or null. **Returns:**an IIOMetadata object, or null if the plug-in does not provide metadata encoding capabilities.

### convertImageMetadata

public abstract [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) **convertImageMetadata**([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) inData,  
 [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)

**Description copied from interface:** [**ImageTranscoder**](http://docs.google.com/javax/imageio/ImageTranscoder.html#convertImageMetadata(javax.imageio.metadata.IIOMetadata,%20javax.imageio.ImageTypeSpecifier,%20javax.imageio.ImageWriteParam)) Returns an IIOMetadata object that may be used for encoding and optionally modified using its document interfaces or other interfaces specific to the writer plug-in that will be used for encoding.

An optional ImageWriteParam may be supplied for cases where it may affect the structure of the image metadata.

If the supplied ImageWriteParam contains optional setting values not understood by this writer or transcoder, they will be ignored.

**Specified by:**[convertImageMetadata](http://docs.google.com/javax/imageio/ImageTranscoder.html#convertImageMetadata(javax.imageio.metadata.IIOMetadata,%20javax.imageio.ImageTypeSpecifier,%20javax.imageio.ImageWriteParam)) in interface [ImageTranscoder](http://docs.google.com/javax/imageio/ImageTranscoder.html) **Parameters:**inData - an IIOMetadata object representing image metadata, used to initialize the state of the returned object.imageType - an ImageTypeSpecifier indicating the layout and color information of the image with which the metadata will be associated.param - an ImageWriteParam that will be used to encode the image, or null. **Returns:**an IIOMetadata object, or null if the plug-in does not provide metadata encoding capabilities.

### getNumThumbnailsSupported

public int **getNumThumbnailsSupported**([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param,  
 [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata,  
 [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata)

Returns the number of thumbnails suported by the format being written, given the image type and any additional write parameters and metadata objects that will be used during encoding. A return value of -1 indicates that insufficient information is available.

An ImageWriteParam may optionally be supplied for cases where it may affect thumbnail handling.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

The default implementation returns 0.

**Parameters:**imageType - an ImageTypeSpecifier indicating the type of image to be written, or null.param - the ImageWriteParam that will be used for writing, or null.streamMetadata - an IIOMetadata object that will be used for writing, or null.imageMetadata - an IIOMetadata object that will be used for writing, or null. **Returns:**the number of thumbnails that may be written given the supplied parameters, or -1 if insufficient information is available.

### getPreferredThumbnailSizes

public [Dimension](http://docs.google.com/java/awt/Dimension.html)[] **getPreferredThumbnailSizes**([ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param,  
 [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata,  
 [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata)

Returns an array of Dimensions indicating the legal size ranges for thumbnail images as they will be encoded in the output file or stream. This information is merely advisory; the writer will resize any supplied thumbnails as necessary.

The information is returned as a set of pairs; the first element of a pair contains an (inclusive) minimum width and height, and the second element contains an (inclusive) maximum width and height. Together, each pair defines a valid range of sizes. To specify a fixed size, the same width and height will appear for both elements. A return value of null indicates that the size is arbitrary or unknown.

An ImageWriteParam may optionally be supplied for cases where it may affect thumbnail handling.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

The default implementation returns null.

**Parameters:**imageType - an ImageTypeSpecifier indicating the type of image to be written, or null.param - the ImageWriteParam that will be used for writing, or null.streamMetadata - an IIOMetadata object that will be used for writing, or null.imageMetadata - an IIOMetadata object that will be used for writing, or null. **Returns:**an array of Dimensions with an even length of at least two, or null.

### canWriteRasters

public boolean **canWriteRasters**()

Returns true if the methods that take an IIOImage parameter are capable of dealing with a Raster (as opposed to RenderedImage) source image. If this method returns false, then those methods will throw an UnsupportedOperationException if supplied with an IIOImage containing a Raster.

The default implementation returns false.

**Returns:**true if Raster sources are supported.

### write

public abstract void **write**([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata,  
 [IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Appends a complete image stream containing a single image and associated stream and image metadata and thumbnails to the output. Any necessary header information is included. If the output is an ImageOutputStream, its existing contents prior to the current seek position are not affected, and need not be readable or writable.

The output must have been set beforehand using the setOutput method.

Stream metadata may optionally be supplied; if it is null, default stream metadata will be used.

If canWriteRasters returns true, the IIOImage may contain a Raster source. Otherwise, it must contain a RenderedImage source.

The supplied thumbnails will be resized if needed, and any thumbnails in excess of the supported number will be ignored. If the format requires additional thumbnails that are not provided, the writer should generate them internally.

An ImageWriteParam may optionally be supplied to control the writing process. If param is null, a default write param will be used.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

**Parameters:**streamMetadata - an IIOMetadata object representing stream metadata, or null to use default values.image - an IIOImage object containing an image, thumbnails, and metadata to be written.param - an ImageWriteParam, or null to use a default ImageWriteParam. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if image contains a Raster and canWriteRasters returns false. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if image is null. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### write

public void **write**([IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Appends a complete image stream containing a single image with default metadata and thumbnails to the output. This method is a shorthand for write(null, image, null).

**Parameters:**image - an IIOImage object containing an image, thumbnails, and metadata to be written. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if image is null. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if image contains a Raster and canWriteRasters returns false. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### write

public void **write**([RenderedImage](http://docs.google.com/java/awt/image/RenderedImage.html) image)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Appends a complete image stream consisting of a single image with default metadata and thumbnails to the output. This method is a shorthand for write(null, new IIOImage(image, null, null), null).

**Parameters:**image - a RenderedImage to be written. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if image is null. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### canWriteSequence

public boolean **canWriteSequence**()

Returns true if the writer is able to append an image to an image stream that already contains header information and possibly prior images.

If canWriteSequence returns false, writeToSequence and endWriteSequence will throw an UnsupportedOperationException.

The default implementation returns false.

**Returns:**true if images may be appended sequentially.

### prepareWriteSequence

public void **prepareWriteSequence**([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Prepares a stream to accept a series of subsequent writeToSequence calls, using the provided stream metadata object. The metadata will be written to the stream if it should precede the image data. If the argument is null, default stream metadata is used.

If the output is an ImageOutputStream, the existing contents of the output prior to the current seek position are flushed, and need not be readable or writable. If the format requires that endWriteSequence be able to rewind to patch up the header information, such as for a sequence of images in a single TIFF file, then the metadata written by this method must remain in a writable portion of the stream. Other formats may flush the stream after this method and after each image.

If canWriteSequence returns false, this method will throw an UnsupportedOperationException.

The output must have been set beforehand using either the setOutput method.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**streamMetadata - A stream metadata object, or null. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canWriteSequence returns false. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs writing the stream metadata.

### writeToSequence

public void **writeToSequence**([IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Appends a single image and possibly associated metadata and thumbnails, to the output. If the output is an ImageOutputStream, the existing contents of the output prior to the current seek position may be flushed, and need not be readable or writable, unless the plug-in needs to be able to patch up the header information when endWriteSequence is called (e.g. TIFF).

If canWriteSequence returns false, this method will throw an UnsupportedOperationException.

The output must have been set beforehand using the setOutput method.

prepareWriteSequence must have been called beforehand, or an IllegalStateException is thrown.

If canWriteRasters returns true, the IIOImage may contain a Raster source. Otherwise, it must contain a RenderedImage source.

The supplied thumbnails will be resized if needed, and any thumbnails in excess of the supported number will be ignored. If the format requires additional thumbnails that are not provided, the writer will generate them internally.

An ImageWriteParam may optionally be supplied to control the writing process. If param is null, a default write param will be used.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**image - an IIOImage object containing an image, thumbnails, and metadata to be written.param - an ImageWriteParam, or null to use a default ImageWriteParam. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set, or prepareWriteSequence has not been called. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canWriteSequence returns false. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if image is null. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if image contains a Raster and canWriteRasters returns false. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### endWriteSequence

public void **endWriteSequence**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Completes the writing of a sequence of images begun with prepareWriteSequence. Any stream metadata that should come at the end of the sequence of images is written out, and any header information at the beginning of the sequence is patched up if necessary. If the output is an ImageOutputStream, data through the stream metadata at the end of the sequence are flushed and need not be readable or writable.

If canWriteSequence returns false, this method will throw an UnsupportedOperationException.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set, or prepareWriteSequence has not been called. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canWriteSequence returns false. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### canReplaceStreamMetadata

public boolean **canReplaceStreamMetadata**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns true if it is possible to replace the stream metadata already present in the output.

The default implementation throws an IllegalStateException if the output is null, and otherwise returns false.

**Returns:**true if replacement of stream metadata is allowed. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the query.

### replaceStreamMetadata

public void **replaceStreamMetadata**([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Replaces the stream metadata in the output with new information. If the output is an ImageOutputStream, the prior contents of the stream are examined and possibly edited to make room for the new data. All of the prior contents of the output must be available for reading and writing.

If canReplaceStreamMetadata returns false, an UnsupportedOperationException will be thrown.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**streamMetadata - an IIOMetadata object representing stream metadata, or null to use default values. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if the canReplaceStreamMetadata returns false. modes do not include [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### canReplaceImageMetadata

public boolean **canReplaceImageMetadata**(int imageIndex)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns true if it is possible to replace the image metadata associated with an existing image with index imageIndex. If this method returns false, a call to replaceImageMetadata(imageIndex) will throw an UnsupportedOperationException.

A writer that does not support any image metadata replacement may return false without performing bounds checking on the index.

The default implementation throws an IllegalStateException if the output is null, and otherwise returns false without checking the value of imageIndex.

**Parameters:**imageIndex - the index of the image whose metadata is to be replaced. **Returns:**true if the image metadata of the given image can be replaced. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the writer supports image metadata replacement in general, but imageIndex is less than 0 or greater than the largest available index. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the query.

### replaceImageMetadata

public void **replaceImageMetadata**(int imageIndex,  
 [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Replaces the image metadata associated with an existing image.

If canReplaceImageMetadata(imageIndex) returns false, an UnsupportedOperationException will be thrown.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**imageIndex - the index of the image whose metadata is to be replaced.imageMetadata - an IIOMetadata object representing image metadata, or null. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canReplaceImageMetadata returns false. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if imageIndex is less than 0 or greater than the largest available index. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### canInsertImage

public boolean **canInsertImage**(int imageIndex)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns true if the writer supports the insertion of a new image at the given index. Existing images with indices greater than or equal to the insertion index will have their indices increased by 1. A value for imageIndex of -1 may be used to signify an index one larger than the current largest index.

A writer that does not support any image insertion may return false without performing bounds checking on the index.

The default implementation throws an IllegalStateException if the output is null, and otherwise returns false withour checking the value of imageIndex.

**Parameters:**imageIndex - the index at which the image is to be inserted. **Returns:**true if an image may be inserted at the given index. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the writer supports image insertion in general, but imageIndex is less than -1 or greater than the largest available index. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the query.

### writeInsert

public void **writeInsert**(int imageIndex,  
 [IIOImage](http://docs.google.com/javax/imageio/IIOImage.html) image,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Inserts a new image into an existing image stream. Existing images with an index greater than imageIndex are preserved, and their indices are each increased by 1. A value for imageIndex of -1 may be used to signify an index one larger than the previous largest index; that is, it will cause the image to be logically appended to the end of the sequence. If the output is an ImageOutputStream, the entirety of the stream must be both readable and writeable.

If canInsertImage(imageIndex) returns false, an UnsupportedOperationException will be thrown.

An ImageWriteParam may optionally be supplied to control the writing process. If param is null, a default write param will be used.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**imageIndex - the index at which to write the image.image - an IIOImage object containing an image, thumbnails, and metadata to be written.param - an ImageWriteParam, or null to use a default ImageWriteParam. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canInsertImage(imageIndex) returns false. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if image is null. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if imageIndex is less than -1 or greater than the largest available index. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if image contains a Raster and canWriteRasters returns false. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during writing.

### canRemoveImage

public boolean **canRemoveImage**(int imageIndex)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns true if the writer supports the removal of an existing image at the given index. Existing images with indices greater than the insertion index will have their indices decreased by 1.

A writer that does not support any image removal may return false without performing bounds checking on the index.

The default implementation throws an IllegalStateException if the output is null, and otherwise returns false without checking the value of imageIndex.

**Parameters:**imageIndex - the index of the image to be removed. **Returns:**true if it is possible to remove the given image. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the writer supports image removal in general, but imageIndex is less than 0 or greater than the largest available index. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the query.

### removeImage

public void **removeImage**(int imageIndex)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Removes an image from the stream.

If canRemoveImage(imageIndex) returns false, an UnsupportedOperationExceptionwill be thrown.

The removal may or may not cause a reduction in the actual file size.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**imageIndex - the index of the image to be removed. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canRemoveImage(imageIndex) returns false. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if imageIndex is less than 0 or greater than the largest available index. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the removal.

### canWriteEmpty

public boolean **canWriteEmpty**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns true if the writer supports the writing of a complete image stream consisting of a single image with undefined pixel values and associated metadata and thumbnails to the output. The pixel values may be defined by future calls to the replacePixels methods. If the output is an ImageOutputStream, its existing contents prior to the current seek position are not affected, and need not be readable or writable.

The default implementation throws an IllegalStateException if the output is null, and otherwise returns false.

**Returns:**true if the writing of complete image stream with contents to be defined later is supported. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the query.

### prepareWriteEmpty

public void **prepareWriteEmpty**([IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) streamMetadata,  
 [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType,  
 int width,  
 int height,  
 [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata,  
 [List](http://docs.google.com/java/util/List.html)<? extends [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html)> thumbnails,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Begins the writing of a complete image stream, consisting of a single image with undefined pixel values and associated metadata and thumbnails, to the output. The pixel values will be defined by future calls to the replacePixels methods. If the output is an ImageOutputStream, its existing contents prior to the current seek position are not affected, and need not be readable or writable.

The writing is not complete until a call to endWriteEmpty occurs. Calls to prepareReplacePixels, replacePixels, and endReplacePixels may occur between calls to prepareWriteEmpty and endWriteEmpty. However, calls to prepareWriteEmpty cannot be nested, and calls to prepareWriteEmpty and prepareInsertEmpty may not be interspersed.

If canWriteEmpty returns false, an UnsupportedOperationException will be thrown.

An ImageWriteParam may optionally be supplied to control the writing process. If param is null, a default write param will be used.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**streamMetadata - an IIOMetadata object representing stream metadata, or null to use default values.imageType - an ImageTypeSpecifier describing the layout of the image.width - the width of the image.height - the height of the image.imageMetadata - an IIOMetadata object representing image metadata, or null.thumbnails - a List of BufferedImage thumbnails for this image, or null.param - an ImageWriteParam, or null to use a default ImageWriteParam. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canWriteEmpty returns false. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareWriteEmpty has been made without a corresponding call to endWriteEmpty. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareInsertEmpty has been made without a corresponding call to endInsertEmpty. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if imageType is null or thumbnails contains null references or objects other than BufferedImages. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if width or height are less than 1. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during writing.

### endWriteEmpty

public void **endWriteEmpty**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Completes the writing of a new image that was begun with a prior call to prepareWriteEmpty.

If canWriteEmpty() returns false, an UnsupportedOperationException will be thrown.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canWriteEmpty(imageIndex) returns false. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareWriteEmpty without a corresponding call to endWriteEmpty has not been made. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareInsertEmpty without a corresponding call to endInsertEmpty has been made. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a call to prepareReiplacePixels has been made without a matching call to endReplacePixels. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during writing.

### canInsertEmpty

public boolean **canInsertEmpty**(int imageIndex)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns true if the writer supports the insertion of a new, empty image at the given index. The pixel values of the image are undefined, and may be specified in pieces using the replacePixels methods. Existing images with indices greater than or equal to the insertion index will have their indices increased by 1. A value for imageIndex of -1 may be used to signify an index one larger than the current largest index.

A writer that does not support insertion of empty images may return false without performing bounds checking on the index.

The default implementation throws an IllegalStateException if the output is null, and otherwise returns false without checking the value of imageIndex.

**Parameters:**imageIndex - the index at which the image is to be inserted. **Returns:**true if an empty image may be inserted at the given index. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the writer supports empty image insertion in general, but imageIndex is less than -1 or greater than the largest available index. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the query.

### prepareInsertEmpty

public void **prepareInsertEmpty**(int imageIndex,  
 [ImageTypeSpecifier](http://docs.google.com/javax/imageio/ImageTypeSpecifier.html) imageType,  
 int width,  
 int height,  
 [IIOMetadata](http://docs.google.com/javax/imageio/metadata/IIOMetadata.html) imageMetadata,  
 [List](http://docs.google.com/java/util/List.html)<? extends [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html)> thumbnails,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Begins the insertion of a new image with undefined pixel values into an existing image stream. Existing images with an index greater than imageIndex are preserved, and their indices are each increased by 1. A value for imageIndex of -1 may be used to signify an index one larger than the previous largest index; that is, it will cause the image to be logically appended to the end of the sequence. If the output is an ImageOutputStream, the entirety of the stream must be both readable and writeable.

The image contents may be supplied later using the replacePixels method. The insertion is not complete until a call to endInsertEmpty occurs. Calls to prepareReplacePixels, replacePixels, and endReplacePixels may occur between calls to prepareInsertEmpty and endInsertEmpty. However, calls to prepareInsertEmpty cannot be nested, and calls to prepareWriteEmpty and prepareInsertEmpty may not be interspersed.

If canInsertEmpty(imageIndex) returns false, an UnsupportedOperationException will be thrown.

An ImageWriteParam may optionally be supplied to control the writing process. If param is null, a default write param will be used.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**imageIndex - the index at which to write the image.imageType - an ImageTypeSpecifier describing the layout of the image.width - the width of the image.height - the height of the image.imageMetadata - an IIOMetadata object representing image metadata, or null.thumbnails - a List of BufferedImage thumbnails for this image, or null.param - an ImageWriteParam, or null to use a default ImageWriteParam. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canInsertEmpty(imageIndex) returns false. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if imageIndex is less than -1 or greater than the largest available index. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareInsertEmpty has been made without a corresponding call to endInsertEmpty. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareWriteEmpty has been made without a corresponding call to endWriteEmpty. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if imageType is null or thumbnails contains null references or objects other than BufferedImages. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if width or height are less than 1. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during writing.

### endInsertEmpty

public void **endInsertEmpty**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Completes the insertion of a new image that was begun with a prior call to prepareInsertEmpty.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canInsertEmpty(imageIndex) returns false. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareInsertEmpty without a corresponding call to endInsertEmpty has not been made. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a previous call to prepareWriteEmpty without a corresponding call to endWriteEmpty has been made. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if a call to prepareReplacePixels has been made without a matching call to endReplacePixels. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during writing.

### canReplacePixels

public boolean **canReplacePixels**(int imageIndex)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns true if the writer allows pixels of the given image to be replaced using the replacePixels methods.

A writer that does not support any pixel replacement may return false without performing bounds checking on the index.

The default implementation throws an IllegalStateException if the output is null, and otherwise returns false without checking the value of imageIndex.

**Parameters:**imageIndex - the index of the image whose pixels are to be replaced. **Returns:**true if the pixels of the given image can be replaced. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the writer supports pixel replacement in general, but imageIndex is less than 0 or greater than the largest available index. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the query.

### prepareReplacePixels

public void **prepareReplacePixels**(int imageIndex,  
 [Rectangle](http://docs.google.com/java/awt/Rectangle.html) region)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Prepares the writer to handle a series of calls to the replacePixels methods. The affected pixel area will be clipped against the supplied

If canReplacePixels returns false, and UnsupportedOperationException will be thrown.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**imageIndex - the index of the image whose pixels are to be replaced.region - a Rectangle that will be used to clip future pixel regions. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canReplacePixels(imageIndex) returns false. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if imageIndex is less than 0 or greater than the largest available index. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if there is a previous call to prepareReplacePixels without a matching call to endReplacePixels (*i.e.*, nesting is not allowed). [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if region is null or has a width or height less than 1. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during the preparation.

### replacePixels

public void **replacePixels**([RenderedImage](http://docs.google.com/java/awt/image/RenderedImage.html) image,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Replaces a portion of an image already present in the output with a portion of the given image. The image data must match, or be convertible to, the image layout of the existing image.

The destination region is specified in the param argument, and will be clipped to the image boundaries and the region supplied to prepareReplacePixels. At least one pixel of the source must not be clipped, or an exception is thrown.

An ImageWriteParam may optionally be supplied to control the writing process. If param is null, a default write param will be used.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

This method may only be called after a call to prepareReplacePixels, or else an IllegalStateException will be thrown.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**image - a RenderedImage containing source pixels.param - an ImageWriteParam, or null to use a default ImageWriteParam. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canReplacePixels(imageIndex) returns false. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if there is no previous call to prepareReplacePixels without a matching call to endReplacePixels. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if any of the following are true:

* image is null.
* param is null.
* the intersected region does not contain at least one pixel.
* the layout of image does not match, or this writer cannot convert it to, the existing image layout.

[IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during writing.

### replacePixels

public void **replacePixels**([Raster](http://docs.google.com/java/awt/image/Raster.html) raster,  
 [ImageWriteParam](http://docs.google.com/javax/imageio/ImageWriteParam.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Replaces a portion of an image already present in the output with a portion of the given Raster. The image data must match, or be convertible to, the image layout of the existing image.

An ImageWriteParam may optionally be supplied to control the writing process. If param is null, a default write param will be used.

The destination region is specified in the param argument, and will be clipped to the image boundaries and the region supplied to prepareReplacePixels. At least one pixel of the source must not be clipped, or an exception is thrown.

If the supplied ImageWriteParam contains optional setting values not supported by this writer (*e.g.* progressive encoding or any format-specific settings), they will be ignored.

This method may only be called after a call to prepareReplacePixels, or else an IllegalStateException will be thrown.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Parameters:**raster - a Raster containing source pixels.param - an ImageWriteParam, or null to use a default ImageWriteParam. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canReplacePixels(imageIndex) returns false. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if there is no previous call to prepareReplacePixels without a matching call to endReplacePixels. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canWriteRasters returns false. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if any of the following are true:

* raster is null.
* param is null.
* the intersected region does not contain at least one pixel.
* the layout of raster does not match, or this writer cannot convert it to, the existing image layout.

[IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during writing.

### endReplacePixels

public void **endReplacePixels**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Terminates a sequence of calls to replacePixels.

If canReplacePixels returns false, and UnsupportedOperationException will be thrown.

The default implementation throws an IllegalStateException if the output is null, and otherwise throws an UnsupportedOperationException.

**Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if the output has not been set. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if canReplacePixels(imageIndex) returns false. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if there is no previous call to prepareReplacePixels without a matching call to endReplacePixels. [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs during writing.

### abort

public void **abort**()

Requests that any current write operation be aborted. The contents of the output following the abort will be undefined.

Writers should call clearAbortRequest at the beginning of each write operation, and poll the value of abortRequested regularly during the write.

### abortRequested

protected boolean **abortRequested**()

Returns true if a request to abort the current write operation has been made since the writer was instantiated or clearAbortRequest was called.

**Returns:**true if the current write operation should be aborted.**See Also:**[abort()](http://docs.google.com/javax/imageio/ImageWriter.html#abort()), [clearAbortRequest()](http://docs.google.com/javax/imageio/ImageWriter.html#clearAbortRequest())

### clearAbortRequest

protected void **clearAbortRequest**()

Clears any previous abort request. After this method has been called, abortRequested will return false.

**See Also:**[abort()](http://docs.google.com/javax/imageio/ImageWriter.html#abort()), [abortRequested()](http://docs.google.com/javax/imageio/ImageWriter.html#abortRequested())

### addIIOWriteWarningListener

public void **addIIOWriteWarningListener**([IIOWriteWarningListener](http://docs.google.com/javax/imageio/event/IIOWriteWarningListener.html) listener)

Adds an IIOWriteWarningListener to the list of registered warning listeners. If listener is null, no exception will be thrown and no action will be taken. Messages sent to the given listener will be localized, if possible, to match the current Locale. If no Locale has been set, warning messages may be localized as the writer sees fit.

**Parameters:**listener - an IIOWriteWarningListener to be registered.**See Also:**[removeIIOWriteWarningListener(javax.imageio.event.IIOWriteWarningListener)](http://docs.google.com/javax/imageio/ImageWriter.html#removeIIOWriteWarningListener(javax.imageio.event.IIOWriteWarningListener))

### removeIIOWriteWarningListener

public void **removeIIOWriteWarningListener**([IIOWriteWarningListener](http://docs.google.com/javax/imageio/event/IIOWriteWarningListener.html) listener)

Removes an IIOWriteWarningListener from the list of registered warning listeners. If the listener was not previously registered, or if listener is null, no exception will be thrown and no action will be taken.

**Parameters:**listener - an IIOWriteWarningListener to be deregistered.**See Also:**[addIIOWriteWarningListener(javax.imageio.event.IIOWriteWarningListener)](http://docs.google.com/javax/imageio/ImageWriter.html#addIIOWriteWarningListener(javax.imageio.event.IIOWriteWarningListener))

### removeAllIIOWriteWarningListeners

public void **removeAllIIOWriteWarningListeners**()

Removes all currently registered IIOWriteWarningListener objects.

The default implementation sets the warningListeners and warningLocales instance variables to null.

### addIIOWriteProgressListener

public void **addIIOWriteProgressListener**([IIOWriteProgressListener](http://docs.google.com/javax/imageio/event/IIOWriteProgressListener.html) listener)

Adds an IIOWriteProgressListener to the list of registered progress listeners. If listener is null, no exception will be thrown and no action will be taken.

**Parameters:**listener - an IIOWriteProgressListener to be registered.**See Also:**[removeIIOWriteProgressListener(javax.imageio.event.IIOWriteProgressListener)](http://docs.google.com/javax/imageio/ImageWriter.html#removeIIOWriteProgressListener(javax.imageio.event.IIOWriteProgressListener))

### removeIIOWriteProgressListener

public void **removeIIOWriteProgressListener**([IIOWriteProgressListener](http://docs.google.com/javax/imageio/event/IIOWriteProgressListener.html) listener)

Removes an IIOWriteProgressListener from the list of registered progress listeners. If the listener was not previously registered, or if listener is null, no exception will be thrown and no action will be taken.

**Parameters:**listener - an IIOWriteProgressListener to be deregistered.**See Also:**[addIIOWriteProgressListener(javax.imageio.event.IIOWriteProgressListener)](http://docs.google.com/javax/imageio/ImageWriter.html#addIIOWriteProgressListener(javax.imageio.event.IIOWriteProgressListener))

### removeAllIIOWriteProgressListeners

public void **removeAllIIOWriteProgressListeners**()

Removes all currently registered IIOWriteProgressListener objects.

The default implementation sets the progressListeners instance variable to null.

### processImageStarted

protected void **processImageStarted**(int imageIndex)

Broadcasts the start of an image write to all registered IIOWriteProgressListeners by calling their imageStarted method. Subclasses may use this method as a convenience.

**Parameters:**imageIndex - the index of the image about to be written.

### processImageProgress

protected void **processImageProgress**(float percentageDone)

Broadcasts the current percentage of image completion to all registered IIOWriteProgressListeners by calling their imageProgress method. Subclasses may use this method as a convenience.

**Parameters:**percentageDone - the current percentage of completion, as a float.

### processImageComplete

protected void **processImageComplete**()

Broadcasts the completion of an image write to all registered IIOWriteProgressListeners by calling their imageComplete method. Subclasses may use this method as a convenience.

### processThumbnailStarted

protected void **processThumbnailStarted**(int imageIndex,  
 int thumbnailIndex)

Broadcasts the start of a thumbnail write to all registered IIOWriteProgressListeners by calling their thumbnailStarted method. Subclasses may use this method as a convenience.

**Parameters:**imageIndex - the index of the image associated with the thumbnail.thumbnailIndex - the index of the thumbnail.

### processThumbnailProgress

protected void **processThumbnailProgress**(float percentageDone)

Broadcasts the current percentage of thumbnail completion to all registered IIOWriteProgressListeners by calling their thumbnailProgress method. Subclasses may use this method as a convenience.

**Parameters:**percentageDone - the current percentage of completion, as a float.

### processThumbnailComplete

protected void **processThumbnailComplete**()

Broadcasts the completion of a thumbnail write to all registered IIOWriteProgressListeners by calling their thumbnailComplete method. Subclasses may use this method as a convenience.

### processWriteAborted

protected void **processWriteAborted**()

Broadcasts that the write has been aborted to all registered IIOWriteProgressListeners by calling their writeAborted method. Subclasses may use this method as a convenience.

### processWarningOccurred

protected void **processWarningOccurred**(int imageIndex,  
 [String](http://docs.google.com/java/lang/String.html) warning)

Broadcasts a warning message to all registered IIOWriteWarningListeners by calling their warningOccurred method. Subclasses may use this method as a convenience.

**Parameters:**imageIndex - the index of the image on which the warning occurred.warning - the warning message. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if warning is null.

### processWarningOccurred

protected void **processWarningOccurred**(int imageIndex,  
 [String](http://docs.google.com/java/lang/String.html) baseName,  
 [String](http://docs.google.com/java/lang/String.html) keyword)

Broadcasts a localized warning message to all registered IIOWriteWarningListeners by calling their warningOccurred method with a string taken from a ResourceBundle. Subclasses may use this method as a convenience.

**Parameters:**imageIndex - the index of the image on which the warning occurred.baseName - the base name of a set of ResourceBundles containing localized warning messages.keyword - the keyword used to index the warning message within the set of ResourceBundles. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if baseName is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if keyword is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if no appropriate ResourceBundle may be located. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the named resource is not found in the located ResourceBundle. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the object retrieved from the ResourceBundle is not a String.

### reset

public void **reset**()

Restores the ImageWriter to its initial state.

The default implementation calls setOutput(null), setLocale(null), removeAllIIOWriteWarningListeners(), removeAllIIOWriteProgressListeners(), and clearAbortRequest.

### dispose

public void **dispose**()

Allows any resources held by this object to be released. The result of calling any other method (other than finalize) subsequent to a call to this method is undefined.

It is important for applications to call this method when they know they will no longer be using this ImageWriter. Otherwise, the writer may continue to hold on to resources indefinitely.

The default implementation of this method in the superclass does nothing. Subclass implementations should ensure that all resources, especially native resources, are released.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ImageWriter.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/ImageWriteParam.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/imageio/ImageWriter.html)    [**NO FRAMES**](http://docs.google.com/ImageWriter.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#1ksv4uv) | [METHOD](#2jxsxqh) |

[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).