
Interface java.awt.image.ImageConsumer

public interface **ImageConsumer**
extends Object

The interface for objects expressing interest in image data through the ImageProducer interfaces. When a consumer is added to an image producer, the producer will deliver all of the data about the image using the method calls defined in this interface.

See Also:
ImageProducer
Version:
1.8 08/29/95
Author:
Jim Graham

Variable Index

Method Index

of a multi−frame animation has been completed, or when an error in loading or producing the image has occured.

- **setColorModel**(ColorModel)
  The ColorModel object which will be used for the majority of the pixels that will be reported using the setPixels method calls.
- **setDimensions**(int, int)
  The dimensions of the source image are reported using the setDimensions method call.
- **setHints**(int)
  The ImageProducer can deliver the pixels in any order, but the ImageConsumer may be able to scale or convert the pixels to the destination ColorModel more efficiently or with higher quality if it knows some information about how the pixels will be delivered up front.
- **setPixels**(int, int, int, int, ColorModel, byte[], int, int)
  The pixels of the image are delivered using one or more calls to the setPixels method.
- **setPixels**(int, int, int, int, ColorModel, int[], int, int)
  The pixels of the image are delivered using one or more calls to the setPixels method.
- **setProperties**(Hashtable)
  The extensible list of properties associated with this image.

Variables

o RANDOMPIXELORDER

```
public final static int RANDOMPIXELORDER
```

The pixels will be delivered in a random order. This tells the ImageConsumer not to use any optimizations that depend on the order of pixel delivery, which should be the default assumption in the absence of any call to the setHints method.

See Also:

setHints

o TOPDOWNLEFTRIGHT

```
public final static int TOPDOWNLEFTRIGHT
```

The pixels will be delivered in top−down, left−to−right order.

See Also:

setHints

o COMPLETESCANLINES

```
public final static int COMPLETESCANLINES
```

The pixels will be delivered in (multiples of) complete scanlines at a time.

See Also:

setHints

- **SINGLEPASS**

```
public final static int SINGLEPASS
```

The pixels will be delivered in a single pass. Each pixel will appear in only one call to any of the setPixels methods. An example of an image format which does not meet this criteria is a progressive JPEG image which defines pixels in multiple passes, each more refined than the previous.

See Also:

setHints

- **SINGLEFRAME**

```
public final static int SINGLEFRAME
```

The image contain a single static image. The pixels will be defined in calls to the setPixels methods and then the imageComplete method will be called with the STATICIMAGEDONE flag after which no more image data will be delivered. An example of an image type which would not meet this criteria would be the output of a video feed, or the representation of a 3D rendering which is being manipulated by the user. The end of each frame in those types of images will be indicated by calling imageComplete with the SINGLEFRAMEDONE flag.

See Also:

setHints, imageComplete

- **IMAGEERROR**

```
public final static int IMAGEERROR
```

An error was encountered while producing the image.

See Also:

imageComplete

- **SINGLEFRAMEDONE**

```
public final static int SINGLEFRAMEDONE
```

One frame of the image is complete but there are more frames to be delivered.

See Also:

imageComplete

- **STATICIMAGEDONE**

```
public final static int STATICIMAGEDONE
```

The image is complete and there are no more pixels or frames to be delivered.

See Also:

imageComplete

Methods

o setDimensions

```
public abstract void setDimensions(int width,
                                   int height)
```

The dimensions of the source image are reported using the setDimensions method call.

o setProperties

```
public abstract void setProperties(Hashtable props)
```

The extensible list of properties associated with this image.

o setColorModel

```
public abstract void setColorModel(ColorModel model)
```

The ColorModel object which will be used for the majority of the pixels that will be reported using the setPixels method calls. Note that each set of pixels delivered using setPixels contains its own ColorModel object, so no assumption should be made that this model will be the only one used in delivering pixel values. A notable case where multiple ColorModel objects may be seen is when looking at a filtered image when the filter determines for each set of pixels that it filters whether the pixels can be sent on untouched, using the original ColorModel, or whether the pixels should be modified (filtered) and passed on using a ColorModel more convenient for the filtering process.

See Also:

ColorModel

o setHints

```
public abstract void setHints(int hintflags)
```

The ImageProducer can deliver the pixels in any order, but the ImageConsumer may be able to scale or convert the pixels to the destination ColorModel more efficiently or with higher quality if it knows some information about how the pixels will be delivered up front. The setHints method should be called before any calls to any of the setPixels methods with a bit mask of hints about the manner in which the pixels will be delivered. If the ImageProducer does not follow the guidelines for the indicated hint, then the results are undefined.

o setPixels

```
public abstract void setPixels(int x,
                               int y,
                               int w,
                               int h,
                               ColorModel model,
                               byte pixels[],
                               int off,
                               int scansize)
```

The pixels of the image are delivered using one or more calls to the setPixels method. Each call specifies the location and size of the rectangle of source pixels that are contained in the array of pixels. The specified ColorModel object should be used to convert the pixels into their corresponding color and alpha components. Pixel (m,n) is stored in the pixels array at index (n * scansize + m + off). The pixels delivered using this method are all stored as bytes.

See Also:

ColorModel

- **setPixels**

```
public abstract void setPixels(int x,
                               int y,
                               int w,
                               int h,
                               ColorModel model,
                               int pixels[],
                               int off,
                               int scansize)
```

The pixels of the image are delivered using one or more calls to the setPixels method. Each call specifies the location and size of the rectangle of source pixels that are contained in the array of pixels. The specified ColorModel object should be used to convert the pixels into their corresponding color and alpha components. Pixel (m,n) is stored in the pixels array at index (n * scansize + m + off). The pixels delivered using this method are all stored as ints.

See Also:

ColorModel

- **imageComplete**

```
public abstract void imageComplete(int status)
```

The imageComplete method is called when the ImageProducer is finished delivering all of the pixels that the source image contains, or when a single frame of a multi–frame animation has been completed, or when an error in loading or producing the image has occured. The ImageConsumer should remove itself from the list of consumers registered with the ImageProducer at this time, unless it is interested in succeeding frames.

See Also:

removeConsumer