| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DirectColorModel.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/awt/image/DataBufferUShort.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/image/FilteredImageSource.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/image/DirectColorModel.html)    [**NO FRAMES**](http://docs.google.com/DirectColorModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#1ksv4uv) |

## **java.awt.image**

Class DirectColorModel

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

**All Implemented Interfaces:** [Transparency](http://docs.google.com/java/awt/Transparency.html)

public class **DirectColorModel**extends [PackedColorModel](http://docs.google.com/java/awt/image/PackedColorModel.html)

The DirectColorModel class is a ColorModel class that works with pixel values that represent RGB color and alpha information as separate samples and that pack all samples for a single pixel into a single int, short, or byte quantity. This class can be used only with ColorSpaces of type ColorSpace.TYPE\_RGB. In addition, for each component of the ColorSpace, the minimum normalized component value obtained via the getMinValue() method of ColorSpace must be 0.0, and the maximum value obtained via the getMaxValue() method must be 1.0 (these min/max values are typical for RGB spaces). There must be three color samples in the pixel values and there can be a single alpha sample. For those methods that use a primitive array pixel representation of type transferType, the array length is always one. The transfer types supported are DataBuffer.TYPE\_BYTE, DataBuffer.TYPE\_USHORT, and DataBuffer.TYPE\_INT. Color and alpha samples are stored in the single element of the array in bits indicated by bit masks. Each bit mask must be contiguous and masks must not overlap. The same masks apply to the single int pixel representation used by other methods. The correspondence of masks and color/alpha samples is as follows:

* Masks are identified by indices running from 0 through 2 if no alpha is present, or 3 if an alpha is present.
* The first three indices refer to color samples; index 0 corresponds to red, index 1 to green, and index 2 to blue.
* Index 3 corresponds to the alpha sample, if present.

The translation from pixel values to color/alpha components for display or processing purposes is a one-to-one correspondence of samples to components. A DirectColorModel is typically used with image data which uses masks to define packed samples. For example, a DirectColorModel can be used in conjunction with a SinglePixelPackedSampleModel to construct a [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html). Normally the masks used by the [SampleModel](http://docs.google.com/java/awt/image/SampleModel.html) and the ColorModel would be the same. However, if they are different, the color interpretation of pixel data will be done according to the masks of the ColorModel.

A single int pixel representation is valid for all objects of this class, since it is always possible to represent pixel values used with this class in a single int. Therefore, methods which use this representation will not throw an IllegalArgumentException due to an invalid pixel value.

This color model is similar to an X11 TrueColor visual. The default RGB ColorModel specified by the [getRGBdefault](http://docs.google.com/java/awt/image/ColorModel.html#getRGBdefault()) method is a DirectColorModel with the following parameters:

Number of bits: 32  
 Red mask: 0x00ff0000  
 Green mask: 0x0000ff00  
 Blue mask: 0x000000ff  
 Alpha mask: 0xff000000  
 Color space: sRGB  
 isAlphaPremultiplied: False  
 Transparency: Transparency.TRANSLUCENT  
 transferType: DataBuffer.TYPE\_INT

Many of the methods in this class are final. This is because the underlying native graphics code makes assumptions about the layout and operation of this class and those assumptions are reflected in the implementations of the methods here that are marked final. You can subclass this class for other reasons, but you cannot override or modify the behavior of those methods.

**See Also:**[ColorModel](http://docs.google.com/java/awt/image/ColorModel.html), [ColorSpace](http://docs.google.com/java/awt/color/ColorSpace.html), [SinglePixelPackedSampleModel](http://docs.google.com/java/awt/image/SinglePixelPackedSampleModel.html), [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html), [ColorModel.getRGBdefault()](http://docs.google.com/java/awt/image/ColorModel.html#getRGBdefault())

| **Field Summary** | |
| --- | --- |

| **Fields inherited from class java.awt.image.**[**ColorModel**](http://docs.google.com/java/awt/image/ColorModel.html) |
| --- |
| [pixel\_bits](http://docs.google.com/java/awt/image/ColorModel.html#pixel_bits), [transferType](http://docs.google.com/java/awt/image/ColorModel.html#transferType) |

| **Fields inherited from interface java.awt.**[**Transparency**](http://docs.google.com/java/awt/Transparency.html) |
| --- |
| [BITMASK](http://docs.google.com/java/awt/Transparency.html#BITMASK), [OPAQUE](http://docs.google.com/java/awt/Transparency.html#OPAQUE), [TRANSLUCENT](http://docs.google.com/java/awt/Transparency.html#TRANSLUCENT) |

| **Constructor Summary** | |
| --- | --- |
| [**DirectColorModel**](http://docs.google.com/java/awt/image/DirectColorModel.html#DirectColorModel(java.awt.color.ColorSpace,%20int,%20int,%20int,%20int,%20int,%20boolean,%20int))([ColorSpace](http://docs.google.com/java/awt/color/ColorSpace.html) space, int bits, int rmask, int gmask, int bmask, int amask, boolean isAlphaPremultiplied, int transferType)            Constructs a DirectColorModel from the specified parameters. |
| [**DirectColorModel**](http://docs.google.com/java/awt/image/DirectColorModel.html#DirectColorModel(int,%20int,%20int,%20int))(int bits, int rmask, int gmask, int bmask)            Constructs a DirectColorModel from the specified masks that indicate which bits in an int pixel representation contain the red, green and blue color samples. |
| [**DirectColorModel**](http://docs.google.com/java/awt/image/DirectColorModel.html#DirectColorModel(int,%20int,%20int,%20int,%20int))(int bits, int rmask, int gmask, int bmask, int amask)            Constructs a DirectColorModel from the specified masks that indicate which bits in an int pixel representation contain the red, green and blue color samples and the alpha sample, if present. |

| **Method Summary** | |
| --- | --- |
| [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) | [**coerceData**](http://docs.google.com/java/awt/image/DirectColorModel.html#coerceData(java.awt.image.WritableRaster,%20boolean))([WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) raster, boolean isAlphaPremultiplied)            Forces the raster data to match the state specified in the isAlphaPremultiplied variable, assuming the data is currently correctly described by this ColorModel. |
| [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) | [**createCompatibleWritableRaster**](http://docs.google.com/java/awt/image/DirectColorModel.html#createCompatibleWritableRaster(int,%20int))(int w, int h)            Creates a WritableRaster with the specified width and height that has a data layout (SampleModel) compatible with this ColorModel. |
| int | [**getAlpha**](http://docs.google.com/java/awt/image/DirectColorModel.html#getAlpha(int))(int pixel)            Returns the alpha component for the specified pixel, scaled from 0 to 255. |
| int | [**getAlpha**](http://docs.google.com/java/awt/image/DirectColorModel.html#getAlpha(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) inData)            Returns the alpha component for the specified pixel, scaled from 0 to 255. |
| int | [**getAlphaMask**](http://docs.google.com/java/awt/image/DirectColorModel.html#getAlphaMask())()            Returns the mask indicating which bits in an int pixel representation contain the alpha component. |
| int | [**getBlue**](http://docs.google.com/java/awt/image/DirectColorModel.html#getBlue(int))(int pixel)            Returns the blue color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. |
| int | [**getBlue**](http://docs.google.com/java/awt/image/DirectColorModel.html#getBlue(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) inData)            Returns the blue color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. |
| int | [**getBlueMask**](http://docs.google.com/java/awt/image/DirectColorModel.html#getBlueMask())()            Returns the mask indicating which bits in an int pixel representation contain the blue color component. |
| int[] | [**getComponents**](http://docs.google.com/java/awt/image/DirectColorModel.html#getComponents(int,%20int%5B%5D,%20int))(int pixel, int[] components, int offset)            Returns an array of unnormalized color/alpha components given a pixel in this ColorModel. |
| int[] | [**getComponents**](http://docs.google.com/java/awt/image/DirectColorModel.html#getComponents(java.lang.Object,%20int%5B%5D,%20int))([Object](http://docs.google.com/java/lang/Object.html) pixel, int[] components, int offset)            Returns an array of unnormalized color/alpha components given a pixel in this ColorModel. |
| int | [**getDataElement**](http://docs.google.com/java/awt/image/DirectColorModel.html#getDataElement(int%5B%5D,%20int))(int[] components, int offset)            Returns a pixel value represented as an int in this ColorModel, given an array of unnormalized color/alpha components. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getDataElements**](http://docs.google.com/java/awt/image/DirectColorModel.html#getDataElements(int%5B%5D,%20int,%20java.lang.Object))(int[] components, int offset, [Object](http://docs.google.com/java/lang/Object.html) obj)            Returns a data element array representation of a pixel in this ColorModel, given an array of unnormalized color/alpha components. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getDataElements**](http://docs.google.com/java/awt/image/DirectColorModel.html#getDataElements(int,%20java.lang.Object))(int rgb, [Object](http://docs.google.com/java/lang/Object.html) pixel)            Returns a data element array representation of a pixel in this ColorModel, given an integer pixel representation in the default RGB color model. |
| int | [**getGreen**](http://docs.google.com/java/awt/image/DirectColorModel.html#getGreen(int))(int pixel)            Returns the green color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. |
| int | [**getGreen**](http://docs.google.com/java/awt/image/DirectColorModel.html#getGreen(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) inData)            Returns the green color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. |
| int | [**getGreenMask**](http://docs.google.com/java/awt/image/DirectColorModel.html#getGreenMask())()            Returns the mask indicating which bits in an int pixel representation contain the green color component. |
| int | [**getRed**](http://docs.google.com/java/awt/image/DirectColorModel.html#getRed(int))(int pixel)            Returns the red color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. |
| int | [**getRed**](http://docs.google.com/java/awt/image/DirectColorModel.html#getRed(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) inData)            Returns the red color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. |
| int | [**getRedMask**](http://docs.google.com/java/awt/image/DirectColorModel.html#getRedMask())()            Returns the mask indicating which bits in an int pixel representation contain the red color component. |
| int | [**getRGB**](http://docs.google.com/java/awt/image/DirectColorModel.html#getRGB(int))(int pixel)            Returns the color/alpha components of the pixel in the default RGB color model format. |
| int | [**getRGB**](http://docs.google.com/java/awt/image/DirectColorModel.html#getRGB(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) inData)            Returns the color/alpha components for the specified pixel in the default RGB color model format. |
| boolean | [**isCompatibleRaster**](http://docs.google.com/java/awt/image/DirectColorModel.html#isCompatibleRaster(java.awt.image.Raster))([Raster](http://docs.google.com/java/awt/image/Raster.html) raster)            Returns true if raster is compatible with this ColorModel and false if it is not. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/awt/image/DirectColorModel.html#toString())()            Returns a String that represents this DirectColorModel. |

| **Methods inherited from class java.awt.image.**[**PackedColorModel**](http://docs.google.com/java/awt/image/PackedColorModel.html) |
| --- |
| [createCompatibleSampleModel](http://docs.google.com/java/awt/image/PackedColorModel.html#createCompatibleSampleModel(int,%20int)), [equals](http://docs.google.com/java/awt/image/PackedColorModel.html#equals(java.lang.Object)), [getAlphaRaster](http://docs.google.com/java/awt/image/PackedColorModel.html#getAlphaRaster(java.awt.image.WritableRaster)), [getMask](http://docs.google.com/java/awt/image/PackedColorModel.html#getMask(int)), [getMasks](http://docs.google.com/java/awt/image/PackedColorModel.html#getMasks()), [isCompatibleSampleModel](http://docs.google.com/java/awt/image/PackedColorModel.html#isCompatibleSampleModel(java.awt.image.SampleModel)) |

| **Methods inherited from class java.awt.image.**[**ColorModel**](http://docs.google.com/java/awt/image/ColorModel.html) |
| --- |
| [finalize](http://docs.google.com/java/awt/image/ColorModel.html#finalize()), [getColorSpace](http://docs.google.com/java/awt/image/ColorModel.html#getColorSpace()), [getComponentSize](http://docs.google.com/java/awt/image/ColorModel.html#getComponentSize()), [getComponentSize](http://docs.google.com/java/awt/image/ColorModel.html#getComponentSize(int)), [getDataElement](http://docs.google.com/java/awt/image/ColorModel.html#getDataElement(float%5B%5D,%20int)), [getDataElements](http://docs.google.com/java/awt/image/ColorModel.html#getDataElements(float%5B%5D,%20int,%20java.lang.Object)), [getNormalizedComponents](http://docs.google.com/java/awt/image/ColorModel.html#getNormalizedComponents(int%5B%5D,%20int,%20float%5B%5D,%20int)), [getNormalizedComponents](http://docs.google.com/java/awt/image/ColorModel.html#getNormalizedComponents(java.lang.Object,%20float%5B%5D,%20int)), [getNumColorComponents](http://docs.google.com/java/awt/image/ColorModel.html#getNumColorComponents()), [getNumComponents](http://docs.google.com/java/awt/image/ColorModel.html#getNumComponents()), [getPixelSize](http://docs.google.com/java/awt/image/ColorModel.html#getPixelSize()), [getRGBdefault](http://docs.google.com/java/awt/image/ColorModel.html#getRGBdefault()), [getTransferType](http://docs.google.com/java/awt/image/ColorModel.html#getTransferType()), [getTransparency](http://docs.google.com/java/awt/image/ColorModel.html#getTransparency()), [getUnnormalizedComponents](http://docs.google.com/java/awt/image/ColorModel.html#getUnnormalizedComponents(float%5B%5D,%20int,%20int%5B%5D,%20int)), [hasAlpha](http://docs.google.com/java/awt/image/ColorModel.html#hasAlpha()), [hashCode](http://docs.google.com/java/awt/image/ColorModel.html#hashCode()), [isAlphaPremultiplied](http://docs.google.com/java/awt/image/ColorModel.html#isAlphaPremultiplied()) |

| **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()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [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)) |

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

### DirectColorModel

public **DirectColorModel**(int bits,  
 int rmask,  
 int gmask,  
 int bmask)

Constructs a DirectColorModel from the specified masks that indicate which bits in an int pixel representation contain the red, green and blue color samples. As pixel values do not contain alpha information, all pixels are treated as opaque, which means that alpha = 1.0. All of the bits in each mask must be contiguous and fit in the specified number of least significant bits of an int pixel representation. The ColorSpace is the default sRGB space. The transparency value is Transparency.OPAQUE. The transfer type is the smallest of DataBuffer.TYPE\_BYTE, DataBuffer.TYPE\_USHORT, or DataBuffer.TYPE\_INT that can hold a single pixel.

**Parameters:**bits - the number of bits in the pixel values; for example, the sum of the number of bits in the masks.rmask - specifies a mask indicating which bits in an integer pixel contain the red componentgmask - specifies a mask indicating which bits in an integer pixel contain the green componentbmask - specifies a mask indicating which bits in an integer pixel contain the blue component

### DirectColorModel

public **DirectColorModel**(int bits,  
 int rmask,  
 int gmask,  
 int bmask,  
 int amask)

Constructs a DirectColorModel from the specified masks that indicate which bits in an int pixel representation contain the red, green and blue color samples and the alpha sample, if present. If amask is 0, pixel values do not contain alpha information and all pixels are treated as opaque, which means that alpha = 1.0. All of the bits in each mask must be contiguous and fit in the specified number of least significant bits of an int pixel representation. Alpha, if present, is not premultiplied. The ColorSpace is the default sRGB space. The transparency value is Transparency.OPAQUE if no alpha is present, or Transparency.TRANSLUCENT otherwise. The transfer type is the smallest of DataBuffer.TYPE\_BYTE, DataBuffer.TYPE\_USHORT, or DataBuffer.TYPE\_INT that can hold a single pixel.

**Parameters:**bits - the number of bits in the pixel values; for example, the sum of the number of bits in the masks.rmask - specifies a mask indicating which bits in an integer pixel contain the red componentgmask - specifies a mask indicating which bits in an integer pixel contain the green componentbmask - specifies a mask indicating which bits in an integer pixel contain the blue componentamask - specifies a mask indicating which bits in an integer pixel contain the alpha component

### DirectColorModel

public **DirectColorModel**([ColorSpace](http://docs.google.com/java/awt/color/ColorSpace.html) space,  
 int bits,  
 int rmask,  
 int gmask,  
 int bmask,  
 int amask,  
 boolean isAlphaPremultiplied,  
 int transferType)

Constructs a DirectColorModel from the specified parameters. Color components are in the specified ColorSpace, which must be of type ColorSpace.TYPE\_RGB and have minimum normalized component values which are all 0.0 and maximum values which are all 1.0. The masks specify which bits in an int pixel representation contain the red, green and blue color samples and the alpha sample, if present. If amask is 0, pixel values do not contain alpha information and all pixels are treated as opaque, which means that alpha = 1.0. All of the bits in each mask must be contiguous and fit in the specified number of least significant bits of an int pixel representation. If there is alpha, the boolean isAlphaPremultiplied specifies how to interpret color and alpha samples in pixel values. If the boolean is true, color samples are assumed to have been multiplied by the alpha sample. The transparency value is Transparency.OPAQUE, if no alpha is present, or Transparency.TRANSLUCENT otherwise. The transfer type is the type of primitive array used to represent pixel values and must be one of DataBuffer.TYPE\_BYTE, DataBuffer.TYPE\_USHORT, or DataBuffer.TYPE\_INT.

**Parameters:**space - the specified ColorSpacebits - the number of bits in the pixel values; for example, the sum of the number of bits in the masks.rmask - specifies a mask indicating which bits in an integer pixel contain the red componentgmask - specifies a mask indicating which bits in an integer pixel contain the green componentbmask - specifies a mask indicating which bits in an integer pixel contain the blue componentamask - specifies a mask indicating which bits in an integer pixel contain the alpha componentisAlphaPremultiplied - true if color samples are premultiplied by the alpha sample; false otherwisetransferType - the type of array used to represent pixel values **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if space is not a TYPE\_RGB space or if the min/max normalized component values are not 0.0/1.0.

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

### getRedMask

public final int **getRedMask**()

Returns the mask indicating which bits in an int pixel representation contain the red color component.

**Returns:**the mask, which indicates which bits of the int pixel representation contain the red color sample.

### getGreenMask

public final int **getGreenMask**()

Returns the mask indicating which bits in an int pixel representation contain the green color component.

**Returns:**the mask, which indicates which bits of the int pixel representation contain the green color sample.

### getBlueMask

public final int **getBlueMask**()

Returns the mask indicating which bits in an int pixel representation contain the blue color component.

**Returns:**the mask, which indicates which bits of the int pixel representation contain the blue color sample.

### getAlphaMask

public final int **getAlphaMask**()

Returns the mask indicating which bits in an int pixel representation contain the alpha component.

**Returns:**the mask, which indicates which bits of the int pixel representation contain the alpha sample.

### getRed

public final int **getRed**(int pixel)

Returns the red color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. A color conversion is done if necessary. The pixel value is specified as an int. The returned value is a non pre-multiplied value. Thus, if the alpha is premultiplied, this method divides it out before returning the value. If the alpha value is 0, for example, the red value is 0.

**Specified by:**[getRed](http://docs.google.com/java/awt/image/ColorModel.html#getRed(int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**pixel - the specified pixel **Returns:**the red color component for the specified pixel, from 0 to 255 in the sRGB ColorSpace.

### getGreen

public final int **getGreen**(int pixel)

Returns the green color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. A color conversion is done if necessary. The pixel value is specified as an int. The returned value is a non pre-multiplied value. Thus, if the alpha is premultiplied, this method divides it out before returning the value. If the alpha value is 0, for example, the green value is 0.

**Specified by:**[getGreen](http://docs.google.com/java/awt/image/ColorModel.html#getGreen(int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**pixel - the specified pixel **Returns:**the green color component for the specified pixel, from 0 to 255 in the sRGB ColorSpace.

### getBlue

public final int **getBlue**(int pixel)

Returns the blue color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. A color conversion is done if necessary. The pixel value is specified as an int. The returned value is a non pre-multiplied value. Thus, if the alpha is premultiplied, this method divides it out before returning the value. If the alpha value is 0, for example, the blue value is 0.

**Specified by:**[getBlue](http://docs.google.com/java/awt/image/ColorModel.html#getBlue(int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**pixel - the specified pixel **Returns:**the blue color component for the specified pixel, from 0 to 255 in the sRGB ColorSpace.

### getAlpha

public final int **getAlpha**(int pixel)

Returns the alpha component for the specified pixel, scaled from 0 to 255. The pixel value is specified as an int.

**Specified by:**[getAlpha](http://docs.google.com/java/awt/image/ColorModel.html#getAlpha(int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**pixel - the specified pixel **Returns:**the value of the alpha component of pixel from 0 to 255.

### getRGB

public final int **getRGB**(int pixel)

Returns the color/alpha components of the pixel in the default RGB color model format. A color conversion is done if necessary. The pixel value is specified as an int. The returned value is in a non pre-multiplied format. Thus, if the alpha is premultiplied, this method divides it out of the color components. If the alpha value is 0, for example, the color values are each 0.

**Overrides:**[getRGB](http://docs.google.com/java/awt/image/ColorModel.html#getRGB(int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**pixel - the specified pixel **Returns:**the RGB value of the color/alpha components of the specified pixel.**See Also:**[ColorModel.getRGBdefault()](http://docs.google.com/java/awt/image/ColorModel.html#getRGBdefault())

### getRed

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

Returns the red color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. A color conversion is done if necessary. The pixel value is specified by an array of data elements of type transferType passed in as an object reference. The returned value is a non pre-multiplied value. Thus, if the alpha is premultiplied, this method divides it out before returning the value. If the alpha value is 0, for example, the red value is 0. If inData is not a primitive array of type transferType, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if inData is not large enough to hold a pixel value for this ColorModel. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType. An UnsupportedOperationException is thrown if this transferType is not supported by this ColorModel.

**Overrides:**[getRed](http://docs.google.com/java/awt/image/ColorModel.html#getRed(java.lang.Object)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**inData - the array containing the pixel value **Returns:**the value of the red component of the specified pixel. **Throws:** [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if inData is not large enough to hold a pixel value for this color model [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if inData is not a primitive array of type transferType [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this color model

### getGreen

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

Returns the green color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. A color conversion is done if necessary. The pixel value is specified by an array of data elements of type transferType passed in as an object reference. The returned value is a non pre-multiplied value. Thus, if the alpha is premultiplied, this method divides it out before returning the value. If the alpha value is 0, for example, the green value is 0. If inData is not a primitive array of type transferType, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if inData is not large enough to hold a pixel value for this ColorModel. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType. An UnsupportedOperationException is thrown if this transferType is not supported by this ColorModel.

**Overrides:**[getGreen](http://docs.google.com/java/awt/image/ColorModel.html#getGreen(java.lang.Object)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**inData - the array containing the pixel value **Returns:**the value of the green component of the specified pixel. **Throws:** [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if inData is not large enough to hold a pixel value for this color model [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if inData is not a primitive array of type transferType [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this color model

### getBlue

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

Returns the blue color component for the specified pixel, scaled from 0 to 255 in the default RGB ColorSpace, sRGB. A color conversion is done if necessary. The pixel value is specified by an array of data elements of type transferType passed in as an object reference. The returned value is a non pre-multiplied value. Thus, if the alpha is premultiplied, this method divides it out before returning the value. If the alpha value is 0, for example, the blue value is 0. If inData is not a primitive array of type transferType, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if inData is not large enough to hold a pixel value for this ColorModel. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType. An UnsupportedOperationException is thrown if this transferType is not supported by this ColorModel.

**Overrides:**[getBlue](http://docs.google.com/java/awt/image/ColorModel.html#getBlue(java.lang.Object)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**inData - the array containing the pixel value **Returns:**the value of the blue component of the specified pixel. **Throws:** [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if inData is not large enough to hold a pixel value for this color model [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if inData is not a primitive array of type transferType [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this color model

### getAlpha

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

Returns the alpha component for the specified pixel, scaled from 0 to 255. The pixel value is specified by an array of data elements of type transferType passed in as an object reference. If inData is not a primitive array of type transferType, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if inData is not large enough to hold a pixel value for this ColorModel. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType. If this transferType is not supported, an UnsupportedOperationException is thrown.

**Overrides:**[getAlpha](http://docs.google.com/java/awt/image/ColorModel.html#getAlpha(java.lang.Object)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**inData - the specified pixel **Returns:**the alpha component of the specified pixel, scaled from 0 to 255. **Throws:** ClassCastException - if inData is not a primitive array of type transferType ArrayIndexOutOfBoundsException - if inData is not large enough to hold a pixel value for this ColorModel UnsupportedOperationException - if this tranferType is not supported by this ColorModel

### getRGB

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

Returns the color/alpha components for the specified pixel in the default RGB color model format. A color conversion is done if necessary. The pixel value is specified by an array of data elements of type transferType passed in as an object reference. If inData is not a primitive array of type transferType, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if inData is not large enough to hold a pixel value for this ColorModel. The returned value is in a non pre-multiplied format. Thus, if the alpha is premultiplied, this method divides it out of the color components. If the alpha value is 0, for example, the color values is 0. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType.

**Overrides:**[getRGB](http://docs.google.com/java/awt/image/ColorModel.html#getRGB(java.lang.Object)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**inData - the specified pixel **Returns:**the color and alpha components of the specified pixel. **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this ColorModel**See Also:**[ColorModel.getRGBdefault()](http://docs.google.com/java/awt/image/ColorModel.html#getRGBdefault())

### getDataElements

public [Object](http://docs.google.com/java/lang/Object.html) **getDataElements**(int rgb,  
 [Object](http://docs.google.com/java/lang/Object.html) pixel)

Returns a data element array representation of a pixel in this ColorModel, given an integer pixel representation in the default RGB color model. This array can then be passed to the setDataElements method of a WritableRaster object. If the pixel variable is null, a new array is allocated. If pixel is not null, it must be a primitive array of type transferType; otherwise, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if pixel is not large enough to hold a pixel value for this ColorModel. The pixel array is returned. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType.

**Overrides:**[getDataElements](http://docs.google.com/java/awt/image/ColorModel.html#getDataElements(int,%20java.lang.Object)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**rgb - the integer pixel representation in the default RGB color modelpixel - the specified pixel **Returns:**an array representation of the specified pixel in this ColorModel **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if pixel is not a primitive array of type transferType [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if pixel is not large enough to hold a pixel value for this ColorModel [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this ColorModel**See Also:**[WritableRaster.setDataElements(int, int, java.lang.Object)](http://docs.google.com/java/awt/image/WritableRaster.html#setDataElements(int,%20int,%20java.lang.Object)), [SampleModel.setDataElements(int, int, java.lang.Object, java.awt.image.DataBuffer)](http://docs.google.com/java/awt/image/SampleModel.html#setDataElements(int,%20int,%20java.lang.Object,%20java.awt.image.DataBuffer))

### getComponents

public final int[] **getComponents**(int pixel,  
 int[] components,  
 int offset)

Returns an array of unnormalized color/alpha components given a pixel in this ColorModel. The pixel value is specified as an int. If the components array is null, a new array is allocated. The components array is returned. Color/alpha components are stored in the components array starting at offset, even if the array is allocated by this method. An ArrayIndexOutOfBoundsException is thrown if the components array is not null and is not large enough to hold all the color and alpha components, starting at offset.

**Overrides:**[getComponents](http://docs.google.com/java/awt/image/ColorModel.html#getComponents(int,%20int%5B%5D,%20int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**pixel - the specified pixelcomponents - the array to receive the color and alpha components of the specified pixeloffset - the offset into the components array at which to start storing the color and alpha components **Returns:**an array containing the color and alpha components of the specified pixel starting at the specified offset.

### getComponents

public final int[] **getComponents**([Object](http://docs.google.com/java/lang/Object.html) pixel,  
 int[] components,  
 int offset)

Returns an array of unnormalized color/alpha components given a pixel in this ColorModel. The pixel value is specified by an array of data elements of type transferType passed in as an object reference. If pixel is not a primitive array of type transferType, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if pixel is not large enough to hold a pixel value for this ColorModel. If the components array is null, a new array is allocated. The components array is returned. Color/alpha components are stored in the components array starting at offset, even if the array is allocated by this method. An ArrayIndexOutOfBoundsException is thrown if the components array is not null and is not large enough to hold all the color and alpha components, starting at offset. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType.

**Overrides:**[getComponents](http://docs.google.com/java/awt/image/ColorModel.html#getComponents(java.lang.Object,%20int%5B%5D,%20int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**pixel - the specified pixelcomponents - the array to receive the color and alpha components of the specified pixeloffset - the offset into the components array at which to start storing the color and alpha components **Returns:**an array containing the color and alpha components of the specified pixel starting at the specified offset. **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if pixel is not a primitive array of type transferType [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if pixel is not large enough to hold a pixel value for this ColorModel, or if components is not null and is not large enough to hold all the color and alpha components, starting at offset [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this color model

### createCompatibleWritableRaster

public final [WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) **createCompatibleWritableRaster**(int w,  
 int h)

Creates a WritableRaster with the specified width and height that has a data layout (SampleModel) compatible with this ColorModel.

**Overrides:**[createCompatibleWritableRaster](http://docs.google.com/java/awt/image/ColorModel.html#createCompatibleWritableRaster(int,%20int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**w - the width to apply to the new WritableRasterh - the height to apply to the new WritableRaster **Returns:**a WritableRaster object with the specified width and height. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if w or h is less than or equal to zero**See Also:**[WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html), [SampleModel](http://docs.google.com/java/awt/image/SampleModel.html)

### getDataElement

public int **getDataElement**(int[] components,  
 int offset)

Returns a pixel value represented as an int in this ColorModel, given an array of unnormalized color/alpha components. An ArrayIndexOutOfBoundsException is thrown if the components array is not large enough to hold all the color and alpha components, starting at offset.

**Overrides:**[getDataElement](http://docs.google.com/java/awt/image/ColorModel.html#getDataElement(int%5B%5D,%20int)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**components - an array of unnormalized color and alpha componentsoffset - the index into components at which to begin retrieving the color and alpha components **Returns:**an int pixel value in this ColorModel corresponding to the specified components. **Throws:** ArrayIndexOutOfBoundsException - if the components array is not large enough to hold all of the color and alpha components starting at offset

### getDataElements

public [Object](http://docs.google.com/java/lang/Object.html) **getDataElements**(int[] components,  
 int offset,  
 [Object](http://docs.google.com/java/lang/Object.html) obj)

Returns a data element array representation of a pixel in this ColorModel, given an array of unnormalized color/alpha components. This array can then be passed to the setDataElements method of a WritableRaster object. An ArrayIndexOutOfBoundsException is thrown if the components array is not large enough to hold all the color and alpha components, starting at offset. If the obj variable is null, a new array is allocated. If obj is not null, it must be a primitive array of type transferType; otherwise, a ClassCastException is thrown. An ArrayIndexOutOfBoundsException is thrown if obj is not large enough to hold a pixel value for this ColorModel. Since DirectColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType.

**Overrides:**[getDataElements](http://docs.google.com/java/awt/image/ColorModel.html#getDataElements(int%5B%5D,%20int,%20java.lang.Object)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**components - an array of unnormalized color and alpha componentsoffset - the index into components at which to begin retrieving color and alpha componentsobj - the Object representing an array of color and alpha components **Returns:**an Object representing an array of color and alpha components. **Throws:** ClassCastException - if obj is not a primitive array of type transferType ArrayIndexOutOfBoundsException - if obj is not large enough to hold a pixel value for this ColorModel or the components array is not large enough to hold all of the color and alpha components starting at offset [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this color model**See Also:**[WritableRaster.setDataElements(int, int, java.lang.Object)](http://docs.google.com/java/awt/image/WritableRaster.html#setDataElements(int,%20int,%20java.lang.Object)), [SampleModel.setDataElements(int, int, java.lang.Object, java.awt.image.DataBuffer)](http://docs.google.com/java/awt/image/SampleModel.html#setDataElements(int,%20int,%20java.lang.Object,%20java.awt.image.DataBuffer))

### coerceData

public final [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **coerceData**([WritableRaster](http://docs.google.com/java/awt/image/WritableRaster.html) raster,  
 boolean isAlphaPremultiplied)

Forces the raster data to match the state specified in the isAlphaPremultiplied variable, assuming the data is currently correctly described by this ColorModel. It may multiply or divide the color raster data by alpha, or do nothing if the data is in the correct state. If the data needs to be coerced, this method will also return an instance of this ColorModel with the isAlphaPremultiplied flag set appropriately. This method will throw a UnsupportedOperationException if this transferType is not supported by this ColorModel. Since ColorModel can be subclassed, subclasses inherit the implementation of this method and if they don't override it then they throw an exception if they use an unsupported transferType.

**Overrides:**[coerceData](http://docs.google.com/java/awt/image/ColorModel.html#coerceData(java.awt.image.WritableRaster,%20boolean)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**raster - the WritableRaster dataisAlphaPremultiplied - true if the alpha is premultiplied; false otherwise **Returns:**a ColorModel object that represents the coerced data. **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this transferType is not supported by this color model

### isCompatibleRaster

public boolean **isCompatibleRaster**([Raster](http://docs.google.com/java/awt/image/Raster.html) raster)

Returns true if raster is compatible with this ColorModel and false if it is not.

**Overrides:**[isCompatibleRaster](http://docs.google.com/java/awt/image/ColorModel.html#isCompatibleRaster(java.awt.image.Raster)) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Parameters:**raster - the [Raster](http://docs.google.com/java/awt/image/Raster.html) object to test for compatibility **Returns:**true if raster is compatible with this ColorModel; false otherwise.

### toString

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

Returns a String that represents this DirectColorModel.

**Overrides:**[toString](http://docs.google.com/java/awt/image/ColorModel.html#toString()) in class [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **Returns:**a String representing this DirectColorModel.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DirectColorModel.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/awt/image/DataBufferUShort.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/image/FilteredImageSource.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/image/DirectColorModel.html)    [**NO FRAMES**](http://docs.google.com/DirectColorModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#1ksv4uv) |

[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).