

JAVA BUFFEREDIMAGE CLASS

http://www.tutorialspoint.com/java_dip/java_buffered_image.htm

Copyright © tutorialspoint.com

Java BufferedImage class is a subclass of Image class. It is used to handle and manipulate the image data. A BufferedImage is made of ColorModel of image data. All BufferedImage objects have an upper left corner coordinate of 0, 0.

Constructors

This class supports three types of constructors.

The first constructor constructs a new BufferedImage with a specified ColorModel and Raster.

```
BufferedImage(ColorModel cm, WritableRaster raster,  
boolean isRasterPremultiplied, Hashtable<?,?> properties)
```

The second constructor constructs a BufferedImage of one of the predefined image types.

```
BufferedImage(int width, int height, int imageType)
```

The third constructor constructs a BufferedImage of one of the predefined image types: TYPE_BYTE_BINARY or TYPE_BYTE_INDEXED.

```
BufferedImage(int width, int height, int imageType, IndexColorModel cm)
```

Methods and Description

Sr.No	Methods
1	copyData <i>WritableRaster outRaster</i> It computes an arbitrary rectangular region of the BufferedImage and copies it into a specified WritableRaster.
2	getColorModel It returns object of class ColorModel of an image.
3	getData It returns the image as one large tile.
4	getDataRectanglerect It computes and returns an arbitrary region of the BufferedImage.
5	getGraphics This method returns a Graphics2D, retains backwards compatibility.
6	getHeight

It returns the height of the BufferedImage.

7

getMinX

It returns the minimum x coordinate of this BufferedImage.

8

getMinY

It returns the minimum y coordinate of this BufferedImage.

9

getRGBintx, inty

It returns an integer pixel in the default RGB color model *TYPE_INT_RGB* and default sRGB colorspace.

10

getType

It returns the image type.

Example

The following example demonstrates the use of java BufferedImage class that draw some text on the screen using Graphics Object:

```
import java.awt.Graphics;
import java.awt.Image;
import java.awt.image.BufferedImage;

import javax.swing.JFrame;
import javax.swing.JPanel;

public class Test extends JPanel {

    public void paint(Graphics g) {
        Image img = createImageWithText();
        g.drawImage(img, 20,20,this);
    }

    private Image createImageWithText(){
        BufferedImage bufferedImage = new
BufferedImage(200,200,BufferedImage.TYPE_INT_RGB);
        Graphics g = bufferedImage.getGraphics();

        g.drawString("www.tutorialspoint.com", 20,20);
        g.drawString("www.tutorialspoint.com", 20,40);
        g.drawString("www.tutorialspoint.com", 20,60);
        g.drawString("www.tutorialspoint.com", 20,80);
        g.drawString("www.tutorialspoint.com", 20,100);

        return bufferedImage;
    }

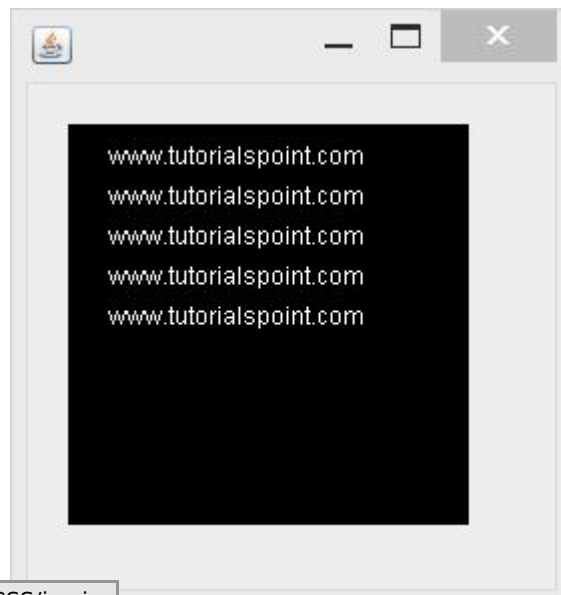
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.getContentPane().add(new Test());

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(200, 200);
    }
}
```

```
    frame.setVisible(true);  
  }  
}
```

Output

When you execute the given code, the following output is seen:



Loading [MathJax]/jax/output/HTML-CSS/jax.js