Borders – createLineBorder()

createLineBorder() creates a Border with a line around it, the line colour must be specified.

Border *line =* ***BorderFactory.createLineBorder****(Color.red);*

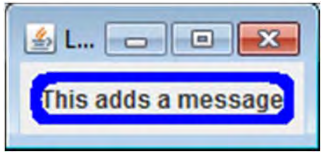
To increase the thickness of the line, specify the number of pixels in the second argument ..

Border *line =* ***BorderFactory.createLineBorder****(Color.blue,5);*

Borders – createLineBorder()

A third parameter in createLineBorder() creates rounded corners for the Border, this can be true or false (default is false)

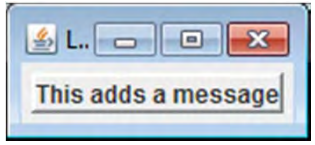
Border *line =* ***BorderFactory.createLineBorder****(Color.blue, 5,* ***true****);*



Borders – createRaisedBevelBorder()

createRaisedBevelBorder() creates an object which protrudes above the rest of the objects

Border *bevel =* ***BorderFactory.createRaisedBevelBorder****();*



For the opposite effect, use createLoweredBevelBorder() ..

Border *line =* ***BorderFactory.createLoweredBevelBorder****();*

Borders – createRaisedSoftBevelBorder()

createRaisedSoftBevelBorder() and the alternative createLoweredSoftBevelBorder() creates objects similar to createRaisedBevelBorder() but with softened corners …

Border *bevel =* ***BorderFactory.createSoftRaisedBevelBorder****();*

or

Border *bevel =* ***BorderFactory.createSoftLoweredBevelBorder****();*

Borders – createBevelBorder()

The problem with the previous Bevel Borders is that neither allows arguments to be passed regarding colour etc, createBevelBorder() allows the following arguments to be passed

```
Border bevel = BorderFactory.createBevelBorder(0);
                    // or BevelBorder.RAISED instead of 0

Border bevel = BorderFactory.createBevelBorder(1);
                    // or BevelBorder.LOWERED instead of 1
```

createSoftBevelBorder() is also available to produce rounded corners.

Borders – createBevelBorder()

Three arguments can be passed to BevelBorder, this first is the type of Bevel as before, the next two are colours for the outer highlight colour and inner highlight colour …

Border *bevel =* ***BorderFactory.createBevelBorder****(1, Color.red, Color.blue);*

A further two colours can be specified, these are for the outer shadow colour and inner shadow colour.

Border *bevel =* ***BorderFactory.createBevelBorder****(1, null, null,*
Color.yellow, Color.white);

Borders – createDashedBorder()

createDashedBorder() allows the Developer to create a broken line around the Component, the simplest usage is to pass the required colour of the line …

Border *bevel* **= *BorderFactory.createDashedBorder(****Color.black****);***

Borders – createDashedBorder()

Two additional arguments can be passed to createDashedBorder(), these specify the length of the dash and the space between the dashes …

Border *bevel* **= BorderFactory.** ***createDashedBorder(Color.black, 4, 2);***

Borders – createDashedBorder()

Finally, five arguments can be passed, these are the colour, the thickness of the dash, the length, the distance between the dashes, and whether to round the corners or not (true or false)

Border *bevel* **= BorderFactory.** ***createDashedBorder***
(Color.*black, 4, 4, 2, true);*

Borders – createStrokeBorder()

Think of a stroke as being a line with different end caps, a BasicStroke is an Object which can be created within createStrokeBorder() or entirely separate. The following must be imported before Strokes can be created …

import java.awt.BasicStroke;

In the following example the Developer is creating a Stroke with a thickness of 3 point …

Borders – createStrokeBorder()


```
import java.awt.BasicStroke;
import javax.swing.BorderFactory;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.border.Border;

public class border1 {

    static JLabel msg = new JLabel("This adds a message");
    static JFrame newframe = new JFrame("Label with border");
    static JPanel pane1 = new JPanel();
    static BasicStroke stroke = new BasicStroke(3);
    static Border labelborder = BorderFactory.createStrokeBorder(stroke);
```

Borders – createStrokeBorder()

Further arguments can be added, this first example accepts three arguments, the first being the thickness, the second the end cap and the third, the join.

Valid values are …

BasicStroke.*CAP_BUTT*
BasicStroke.*CAP_ROUND*
BasicStroke.*CAP_SQUARE*

BasicStroke.*JOIN_MITER*
BasicStroke.*JOIN_BEVEL*
BasicStroke.*JOIN_ROUND*

Borders – createStrokeBorder()

Four arguments are the thickness, cap, join and the length.
Six arguments allow the Developer to specify the gap between the dashes and by using an array the varying lengths of the dashes …

```
public class border1 {

    static float[] dash = {1,2,3};
    static float dash_phase;
    static BasicStroke stroke = new BasicStroke(3
                                                ,BasicStroke.CAP_BUTT
                                                ,BasicStroke.JOIN_MITER
                                                ,6
                                                ,dash
                                                ,dash_phase);
```