```
    c.addMouseListener(new MouseAdapter() {
        //set flag when mouse over
        public void mouseEntered(MouseEvent e) {
            mouse_over = true;
            thisComp.repaint();
        }
        //clear flag when mouse is not over
        public void mouseExited(MouseEvent e) {
            mouse_over = false;
            thisComp.repaint();
        }
    });
}
//--------------
//paint the button
public void paint(Graphics g) {
        super.paint(g);    //first draw the parent button
        //if the mouse is not over the button
        //erase the borders
        if (! mouse_over) {
            Dimension size = super.getSize();
            size.setColor(Color.lightGray);
            g.drawRect(0, 0, size.width-1, size.height-1);
            g.drawLine(size.width-2, 0, size.width-2,
                       size.height-1-);
            g.drawLine(0, size.height-2, size.width-2,
                       size.height-2);
        }
    }
}
```

Using a Decorator

Now that we've written a CoolDecorator class, how do we use it? We simply create an instance of the CoolDecorator and pass it the button that it is to decorate. We can do all of this right in the constructor. Let's consider a simple program that has two CoolButtons and one ordinary JButton. We create the layout as follows:

```
super ("Deco Button");
JPanel jp = new JPanel();
getContentPane().add(jp);

jp.add( new CoolDecorator(new JButton("Cbutton")));
jp.add( new CoolDecorator(new JButton("Dbutton")));
jp.add(Quit = new JButton("Quit"));
Quit.addActionListener(this);
```