

Painting

- When a GUI needs to change its visual appearance it performs a paint operation
- Swing components generally repaint themselves as needed
- Painting code executes on the event-dispatching thread
 - If painting takes a long time, no events will be handled during that time

1/22/01

ICSS235 - Painting

1

Example

```
import javax.swing.*; import java.awt.*;

public class Painting extends JPanel {
    public Painting() {}

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.setColor( Color.yellow ); g.fillOval( 10,10,50,50 );
        g.setColor( Color.black ); g.drawOval( 10,10,50,50 );
    }

    public static void main( String args[] ) {
        JFrame win = new JFrame( "Painting" );
        win.setSize(100, 100);
        win.getContentPane().add( new Painting() );
        win.show();
    }
}
```



1/22/01

ICSS235 - Painting

2

The Graphics Object

- The Graphics object both a context for painting and methods for performing the painting.
- The graphics context consists of state such as the current painting color, the current font, and the current painting area
 - The color and font are initialized to the foreground color and font of the component just before the invocation of paintComponent
- You can ignore the current painting area, if you like

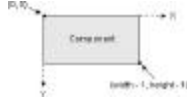
1/22/01

ICSS235 - Painting

3

The Coordinate System

- Each component has its own integer coordinate system
 - Ranging from (0, 0) to (width - 1, height - 1)
 - Each unit represents the size of one pixel



1/22/01

ICSS235 - Painting

4

Borders

- You must take into account the component's size and the size of the component's border
 - A border that paints a one-pixel line around a component changes the top leftmost corner from (0,0) to (1,1) and reduces the width and the height of the painting area by two pixels each
- You get the width and height of a component using its `getWidth` and `getHeight` methods.
- To determine the border size, use the `getInsets` method.

1/22/01

ICSS235 - Painting

5

Example

```
import javax.swing.*; import java.awt.*; import java.awt.Insets.*;

public class Painting extends JPanel {
    public Painting() {}

    public void paintComponent(Graphics g) {
        super.paintComponent(g);

        Insets border = getInsets();
        int width = getWidth() - border.left - border.right;
        int height = getHeight() - border.top - border.bottom;

        int x = ( width / 2 ) - 25 + border.left;
        int y = ( height / 2 ) - 25 + border.top;

        g.setColor( Color.yellow ); g.fillOval( x, y, 50, 50 );
        g.setColor( Color.black ); g.drawOval( x, y, 50, 50 );
    }
} //end Painting
```



ICSS235 - Painting

6

Forcing a Paint

- The `repaint()` method schedules a paint operation for the specified component
 - A version of `repaint()` exists that allows you to specify the area that needs to be repainted
- Typically a component will invoke `repaint()` when it has done something to change its state

1/22/01

ICSS235 - Painting

7

Example

```
import javax.swing.*; import javax.swing.event.*; import java.awt.*;
import java.awt.event.*; import java.awt.Insets.*;

public class Painting extends JPanel {
    private boolean drawn = false;

    private int x; private int y;

    public Painting() {
        addMouseListener(
            new MouseInputAdapter() {
                public void mouseClicked( MouseEvent ev ) {
                    x = ev.getX(); y = ev.getY();
                    repaint();
                }
            }
        );
    }
}
```

1/22/01

ICSS235 - Painting

8

Animation

```
import javax.swing.*; import javax.swing.event.*; import java.awt.*;
import java.awt.event.*; import java.awt.Insets.*;

public class Painting extends JPanel implements ActionListener {
    private boolean drawn = false;
    private int x; private int y;

    private Timer alarm;

    public Painting() {
        alarm = new Timer( 500, this );
        alarm.start();
    }

    public void actionPerformed( ActionEvent ev ) {
        x = x + 10; y = y + 10;
        alarm.restart();
        repaint();
    }
}
```

1/22/01

ICSS235 - Painting

9
