
Classes

Chapter 10 - Lecture Slides

JPanel

JPanel

- Extend the `JPanel` class to create new components
- Can use for
 -
 -
- Good for creating your own components, such as
 - Thermometer
 - Dial
 - Fancy button

JPanel

- Create a separate class that , then add this new class to your applet

```
import java.awt.*;
import javax.swing.*;
public class Smiley extends JPanel
{
}
```

- Add a method **paintComponent** where you can use your drawing methods from chapter 2

```
import java.awt.*;
import javax.swing.*;
public class Smiley extends JPanel
{
    public void paintComponent ( Graphics gr )
    {
        super.paintComponent( gr );
    }
}
```

JPanel

- Add the `Smiley` class in our applet
 - Create a `Smiley` object:

```
Smiley smiles;  
smiles = new Smiley( );
```

- Call the method `setPreferredSize(new Dimension(width, height))` on our `Smiley` object.
If we do not call this method, the default width and height is zero.

```
smiles.setPreferredSize( new Dimension(300, 300) );
```

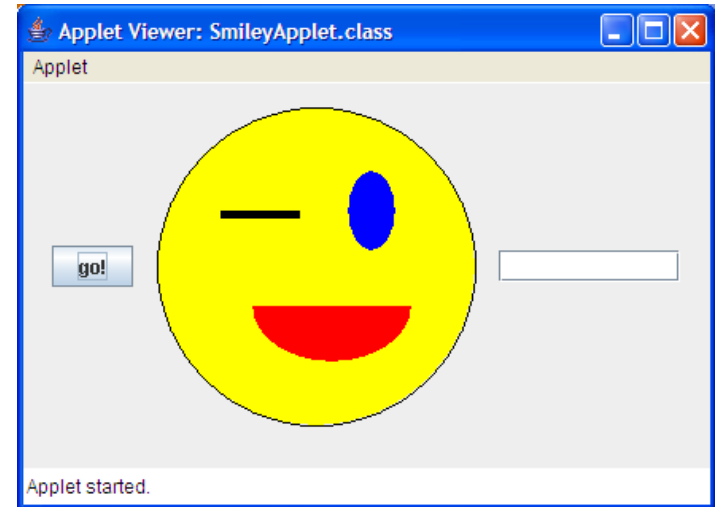
- Lastly, we can add it to the applet.
`add(smiles);`

JPanel

```
import java.awt.*;
import javax.swing.*;
public class SmileyApplet extends JApplet
{
    Smiley smiles;
    JButton button;
    JTextField tf;

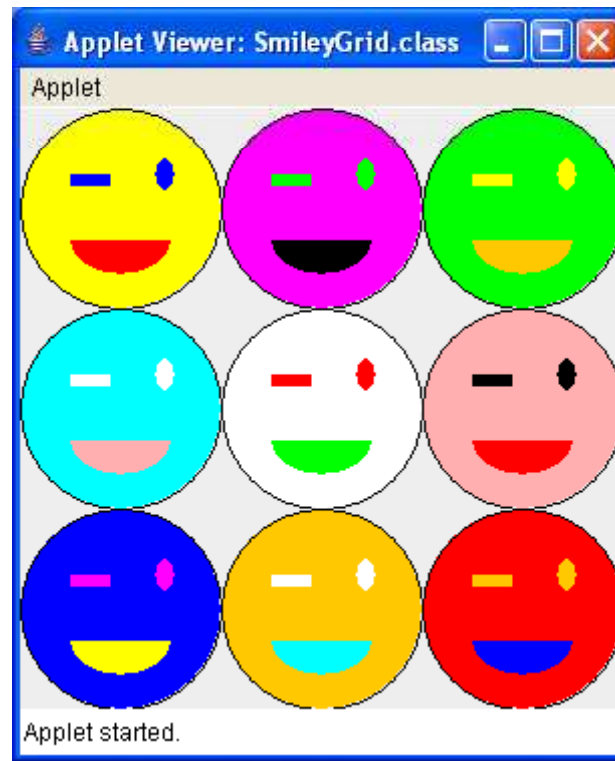
    public void init( )
    {
        // create Smiley object
        smiles = new Smiley( );
        // MUST set the bounds otherwise you won't see it!
        // default is width=0 height=0!!!
        smiles.setPreferredSize( new Dimension( 300,300 ) );

        // create button and textfield
        button = new JButton( "go!" );
        tf = new JTextField(10);
        // add to applet
        setLayout( new FlowLayout( ) );
        add( button );
        add( smiles );
        add( tf );
    }
}
```

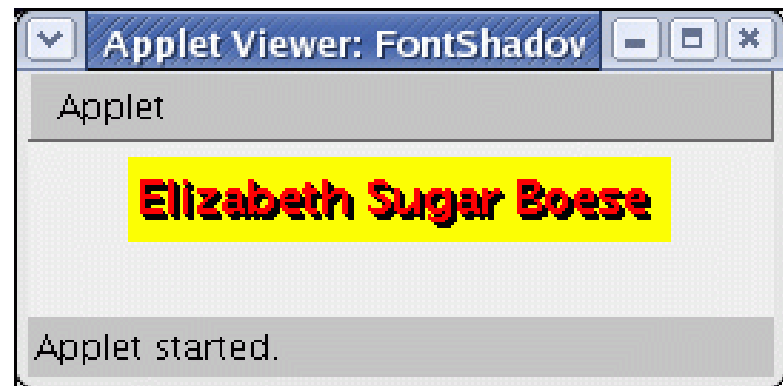
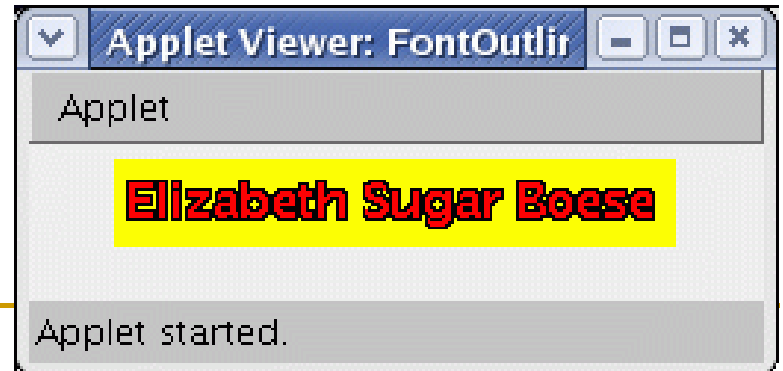


Lecture Exercise:

- Modify the Smiley class to make it easy to create the following applet:



Custom Fonts



Cool Fonts

- Create customized fonts by extending the JPanel class

```
public class FontShadow extends JPanel
{
    // instance variable to hold the text to be displayed
    String txt;
    public FontShadow( String val )
    {
        txt = val;
    }
    public void paintComponent ( Graphics g )
    {
    }
}
```

Shadow Font

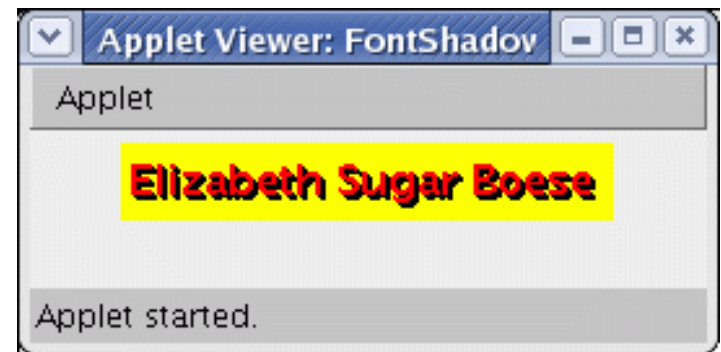
- To create the shadow,

- 1.
- 2.

```
public void paintComponent( Graphics g )
{
    super.paintComponent( g );
    g.setFont( new Font ( "SanSerif", Font.BOLD, 16 ) );
    g.setColor( Color.BLACK );           // can change shadow color
    g.drawString ( txt, 4,21 );
    g.drawString ( txt, 5,22 );
    g.setColor( Color.RED );           // can change text color
    g.drawString ( txt, 3,20 );
}
```

Example using FontShadow

```
import java.awt.*;
import javax.swing.*;
public class FontShadowEx extends JApplet
{
    FontShadow fshad;
    public void init( )
    {
        fshad = new FontShadow( "Elizabeth Sugar Boese" );
        // you have to define it's size
        fshad.setPreferredSize( new Dimension(190,30) );
        setLayout( new FlowLayout( ) );
        add( fshad );
    }
}
```



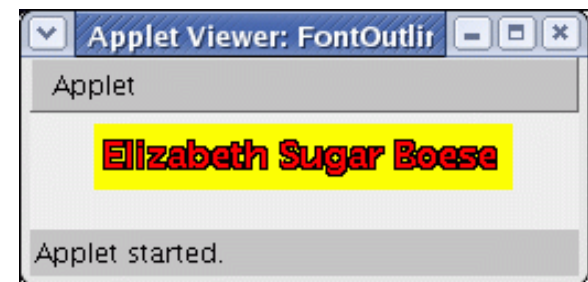
Font Outline



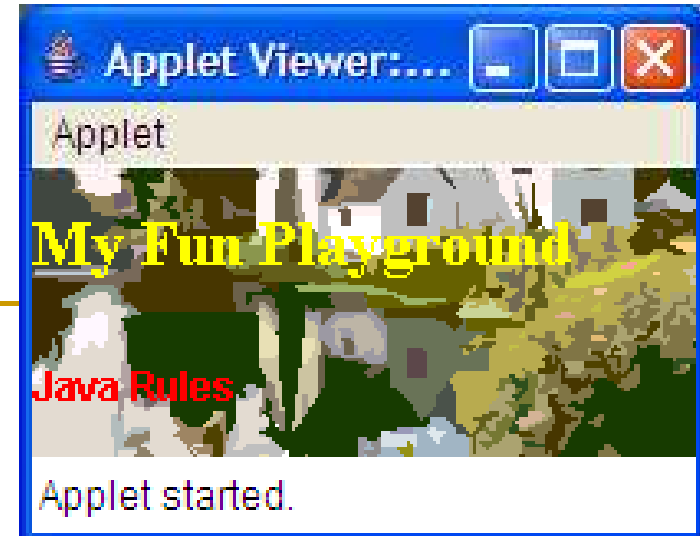
```
public void paintComponent( Graphics g )
{
    super.paintComponent( g );
    g.setFont( new Font ( "SanSerif", Font.BOLD, 16 ) );
    g.setColor( Color.BLACK );           // can change shadow color
    g.drawString ( txt, 3,19 );
    g.drawString ( txt, 3,21 );
    g.drawString ( txt, 5,19 );
    g.drawString ( txt, 5,21 );
    g.setColor( Color.RED );             // can change text color
    g.drawString ( txt, 4,20 );
}
```

Example using FontOutline

```
import java.awt.*;
import javax.swing.*;
public class FontOutlineEx extends JApplet
{
    FontOutline fshad;
    public void init( )
    {
        fshad = new FontOutline( "Elizabeth Sugar Boese" );
        // you have to define it's size
        fshad.setPreferredSize( new Dimension(190,30) );
        setLayout( new FlowLayout( ) );
        add( fshad );
    }
}
```



Background Image



Background Image

- Create a second class "ImgPanel" which extends JPanel and paints the Image in the `paintComponent` method
- Then use this `ImgPanel` as you would normally use `JPanel` in other classes. Add components to it, and the components will appear on top of the image panel.
- Written such that it will center the image within the panel



ImgPanel

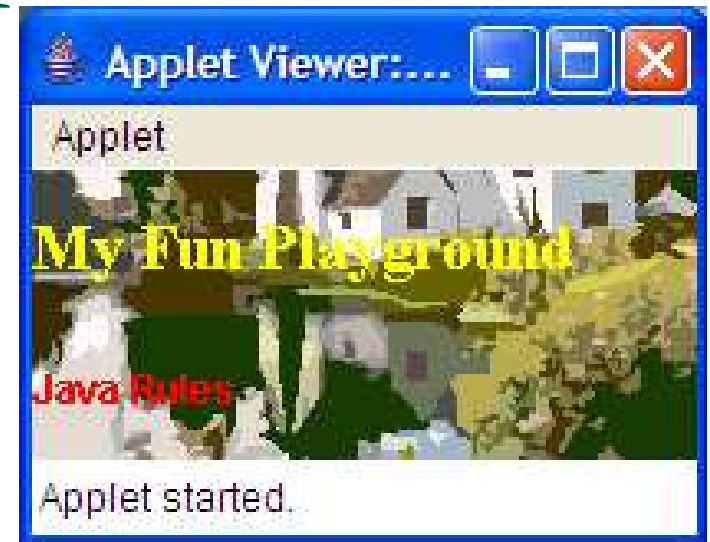
```
public class ImgPanel extends JPanel
{
    ImageIcon icon;
    public ImgPanel( ImageIcon ic )
    {
        icon=ic;
    }
    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        if ( icon != null )
        {
            Image img = icon.getImage();
            int imgWidth = img.getWidth(this);           // find the width of the image
            int panelWidth = this.getWidth( );           // find the width of the panel
            int x = (panelWidth - imgWidth ) / 2;        // calculate x to center the img
            int imgHeight = img.getHeight( this );      // find height of image
            int panelHeight = this.getHeight( );        // find height of panel
            int y = (panelHeight - imgHeight ) / 2;     // calculate y to center the img

            g.drawImage(img,x,y,img.getWidth(this),img.getHeight(this),this);
        }
    }
}
```

Using ImgPanel in an Applet

```
import java.awt.*;
import javax.swing.*;
public class ImgBackground extends JApplet
{
    JLabel myplay, name;
    Image img;
    ImgPanel ipanel;
    public void init( )
    {
        myplay = new JLabel( "My Fun Playground" );
        name = new JLabel( "Java Rules" );
        myplay.setForeground( Color.YELLOW );
        name.setForeground( Color.RED );
        img = getImage( getCodeBase( ), "houseLake.png" );
        ipanel = new ImgPanel( img );

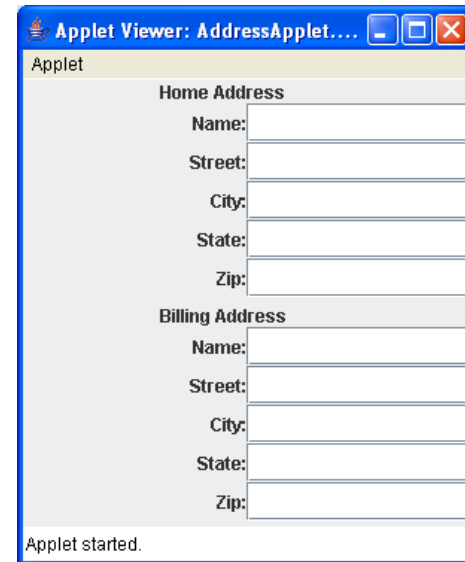
        myplay.setFont( new Font( "Serif", Font.BOLD, 20 ) );
        ipanel.setLayout( new GridLayout( 2,1 ) );
        ipanel.add( myplay );
        ipanel.add( name );
        add( ipanel );
    }
}
```



AddressFields Example

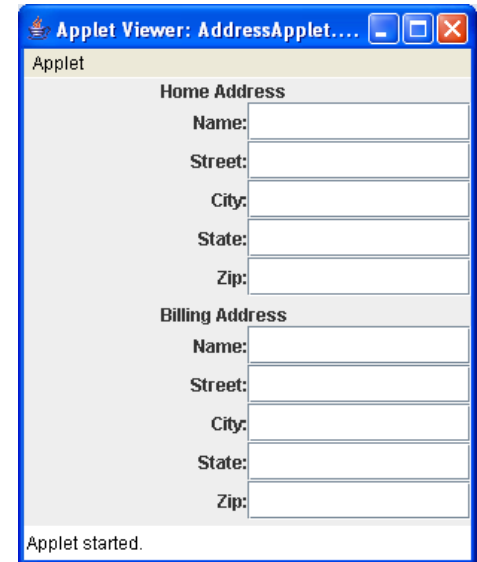
```
import java.awt.*;
import javax.swing.*;
public class AddressFields extends JPanel
{
    // instance variables
    JLabel name, street, city, state, zip;
    JTextField tf_name, tf_street, tf_city, tf_state, tf_zip;

    public AddressFields( ) // constructor
    {
        // initialize the instance variables
        name = new JLabel( "Name:", JLabel.RIGHT );
        street = new JLabel( "Street:", JLabel.RIGHT );
        city = new JLabel( "City:", JLabel.RIGHT );
        state = new JLabel( "State:", JLabel.RIGHT );
        zip = new JLabel( "Zip:", JLabel.RIGHT );
        tf_name = new JTextField( 20 );
        tf_street = new JTextField( 20 );
        tf_city = new JTextField( 20 );
        tf_state = new JTextField( 20 );
        tf_zip = new JTextField( 20 );
        // add to the panel
        setLayout( new GridLayout(5, 2) );
        add( name );    add( tf_name );
        add( street );  add( tf_street );
        add( city );    add( tf_city );
        add( state );   add( tf_state );
        add( zip );     add( tf_zip );
    }
}
```



AddressFields Example

```
import java.awt.*;
import javax.swing.*;
public class AddressApplet extends JApplet
{
    JLabel homeAddress, billingAddress;
    AddressFields home, billing;
    public void init( )
    {
        homeAddress = new JLabel( "Home Address" );
        billingAddress = new JLabel( "Billing Address" );
        home = new AddressFields( );
        billing = new AddressFields( );
        setLayout(new BorderLayout( getContentPane( ), BorderLayout.Y_AXIS ));
        add( homeAddress );
        add( home );
        add( billingAddress );
        add( billing );
    }
}
```



Fancy Button Example

```
import java.awt.*;
import javax.swing.*;
public class FancyButton extends JButton
{
    public FancyButton( Image img, String text )
    {
        setIcon( new ImageIcon(img) );
        setText( text );
        setHorizontalTextPosition( JButton.CENTER );
        setForeground( Color.white );
        setBorderPainted( false );
        setContentAreaFilled( false );
        setFocusable( false );
    }
}
```



```
import java.awt.*;
import javax.swing.*;
public class FancyButtonApplet extends JApplet
{
    Image imge;
    FancyButton home, portfolio, contact;
    public void init( )
    {
        setLayout( new FlowLayout( ) );
        imge = getImage( getCodeBase( ), "buttonGreen.png" );
        home = new FancyButton( imge, "Home" );
        portfolio = new FancyButton( imge, "Portfolio" );
        contact = new FancyButton( imge, "About Us" );
        add( home ); add( portfolio ); add( contact );
    }
}
```

Summary

- JPanel
- Custom Fonts
- Background Image