

# Java Swing



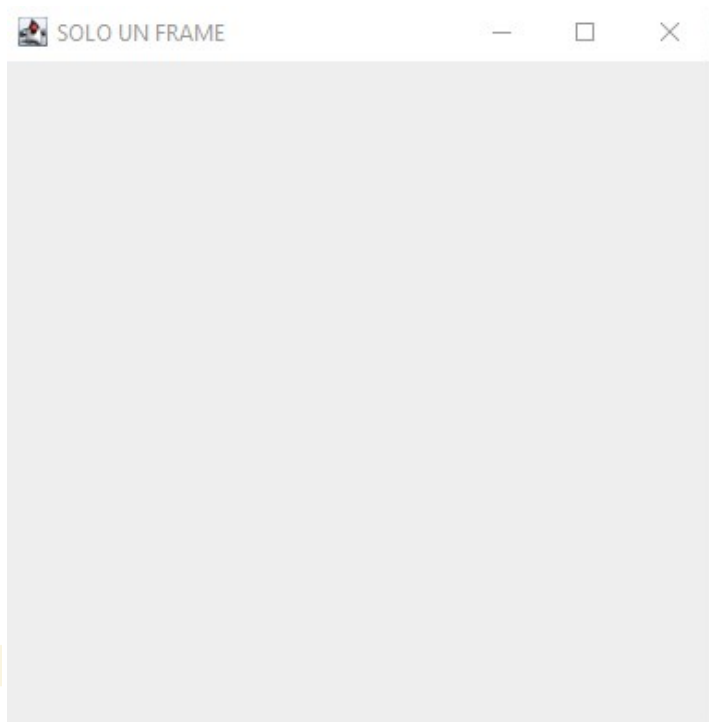
# JFrame

E' il componente principale che contiene gli altri

```
package percopia;
import javax.swing.*;
public class PerCopia {

    public static void main(String[] args) {
        JFrame fin;      // NUOVO OGGETTO FRAME
        fin = new JFrame( title:"SOLO UN FRAME");

        fin.setSize( width:400, height:400); // CARATTERISTICHE DELLA FINESTRA
        fin.setLocation( x:100, y:100);
        fin.setResizable( resizable:true);
        fin.setDefaultCloseOperation( operation:JFrame.EXIT_ON_CLOSE);
        fin.setVisible( b:true);
    }
}
```



# Nel main o come estensione di JFrame

```
// UN FRAME USANDO EREDITARIETA' CLASSE JFrame
package simpleframe;
import javax.swing.*.*;
import java.awt.*.*;

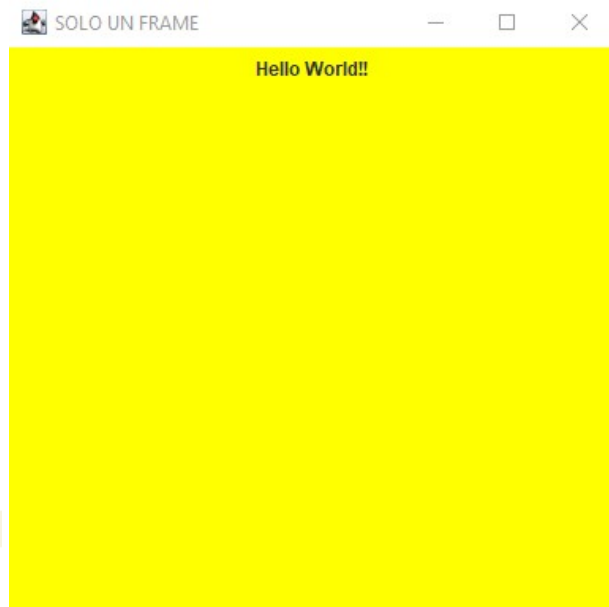
public class SimpleFrame extends JFrame{
    public SimpleFrame(){ // CLASSE SIMPLEFRAME COME ESTENSIONE DI JFrame
        super(); // COSTRUTTORE DI JFrame
        setTitle ( title: "PRIMA FINESTRA CON ESTENSIONE CLASSE"); // METODI
        setSize ( width: 600, height: 500); // SETTA LE DIMENSIONI
        setDefaultCloseOperation( operation: JFrame.EXIT_ON_CLOSE); // CHIUDE IL PROGRAMMA QUANDO SI CHIUDE LA FINESTRA
        setVisible( b: true); // MOSTRA LA FINESTRA
    }

    public static void main(String[] args){
        SimpleFrame fin = new SimpleFrame();

        fin.setLocation( x:100, y:100); // SETTA LA POSIZIONE NELLO SCHERMO
        fin.setResizable( resizable: true); // LA RENDE RIDIMENSIONABILE
        fin.setVisible( b: true); // MOSTRA LA FINESTRA
    }
}
```

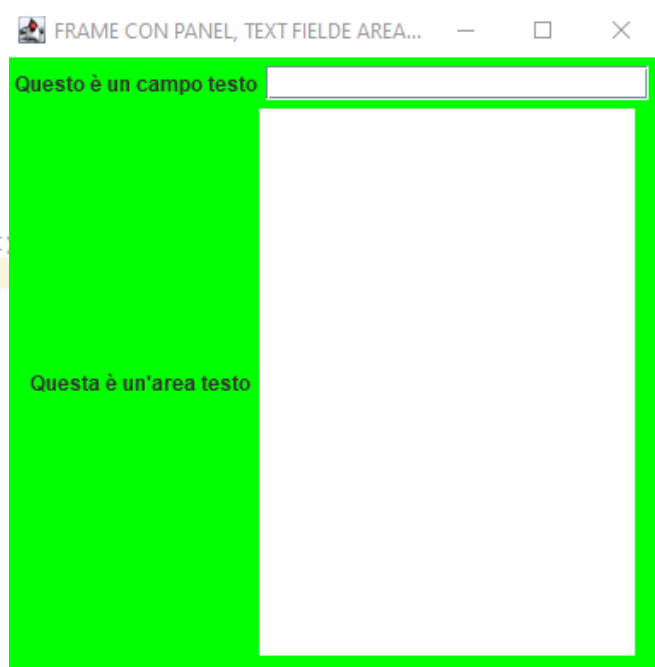
# JPanel e JLabel

```
public class PerCopia {  
  
    public static void main(String[] args) {  
        JFrame fin;        // NUOVO OGGETTO FRAME  
        fin = new JFrame( title: "SOLO UN FRAME");  
  
        JPanel pannello = new JPanel(); // NUOVO OGGETTO PANEL (COSTRUTTORE SENZA PARAMETRI)  
        pannello.setBackground( bg: Color.yellow); // METODO PER CAMBIARE SFONDO AL PANEL  
        pannello.add( new JLabel( text: "Hello World!!" ) ); // AGGIUNGO AL PANEL UNA LABEL  
  
        fin.add( comp: pannello); // AGGIUNGO IL PANEL AL FRAME  
  
        fin.setSize( width: 400, height: 400); // CARATTERISTICHE DELLA FINESTRA  
        fin.setLocation( x: 100, y: 100);  
        fin.setResizable( resizable: true);  
        fin.setDefaultCloseOperation( operation: JFrame.EXIT_ON_CLOSE);  
        fin.setVisible( b: true);  
    }  
}
```



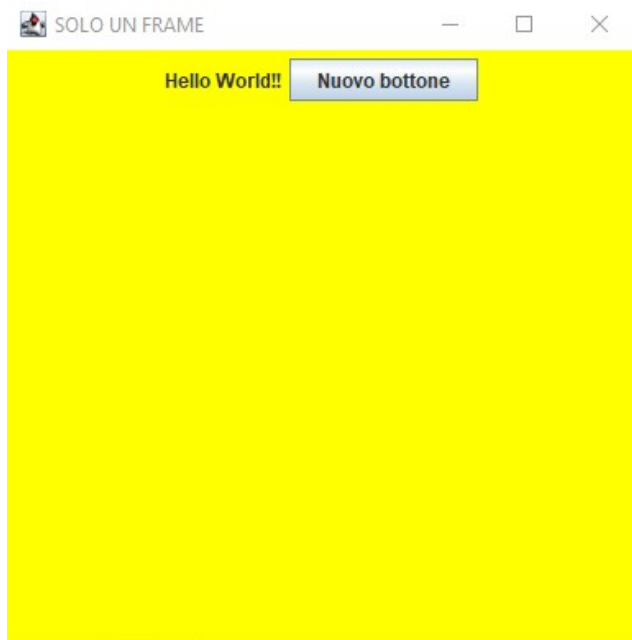
# JtextField, JTextArea

```
public class Finestra_con_elementi2 {  
  
    public static void main(String[] args) {  
        JFrame fin;                                // NUOVO OGGETTO FRAME  
        fin = new JFrame( title: "FRAME CON PANEL, TEXT FIELDE AREA FIELD");  
  
        JPanel pannello = new JPanel();            // NUOVO OGGETTO PANEL (COSTRUTTORE SENZA PARAMETRI)  
        pannello.setBackground( bg: Color.GREEN); // METODO PER CAMBIARE SFONDO AL PANEL  
        pannello.add( new JLabel( text: "Questo è un campo testo" ) ); // AGGIUNGO AL PANEL UNA LABEL  
        JTextField testol = new JTextField( column: 20);  
        pannello.add( comp: testol);                // AGGIUNGO AL PANEL UN CAMPO TESTO  
  
        pannello.add( new JLabel( text: "Questa è un'area testo" ) ); // AGGIUNGO AL PANEL UNA LABEL  
        JTextArea testo2 = new JTextArea( rows: 20, column: 20);  
        pannello.add( comp: testo2);                // AGGIUNGO AL PANEL UN'AREA TESTO  
  
        fin.add( comp: pannello);                    // AGGIUNGO IL PANEL AL FRAME  
  
        fin.setSize( width: 400, height: 400);      // CARATTERISTICHE DELLA FINESTRA  
        fin.setLocation( x: 100, y: 100);  
        fin.setResizable( resizable: true);  
        fin.setDefaultCloseOperation( operation: JFrame.EXIT_ON_CLOSE);  
        fin.setVisible( b: true);  
    }  
}
```



# JButton

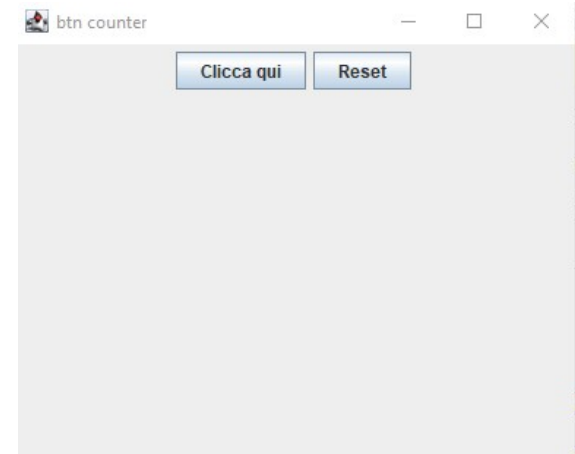
```
public static void main(String[] args) {  
    JFrame fin;        // NUOVO OGGETTO FRAME  
    fin = new JFrame( title: "SOLO UN FRAME");  
  
    JPanel pannello = new JPanel();           // NUOVO OGGETTO PANEL (COSTRUTTORE SENZA PARAMETRI)  
    pannello.setBackground( bg: Color.yellow); // METODO PER CAMBIARE SFONDO AL PANEL  
    pannello.add( new JLabel( text: "Hello World!!" ) ); // AGGIUNGO AL PANEL UNA LABEL  
  
    JButton bot = new JButton( text: "BOTTONE" ); // NUOVO OGGETTO BUTTON (COSTRUTTORE CON PARAMETRO)  
    bot.setText( text: "Nuovo bottone" ); // SETTO UN NUOVO NOME AL PULSANTE  
    pannello.add( comp: bot ); // AGGIUNGO IL PULSANTE AL PANEL  
  
    fin.add( comp: pannello ); // AGGIUNGO IL PANEL AL FRAME  
  
    fin.setSize( width: 400, height: 400 ); // CARATTERISTICHE DELLA FINESTRA  
    fin.setLocation( x: 100, y: 100 );  
    fin.setResizable( resizable: true );  
    fin.setDefaultCloseOperation( operation: JFrame.EXIT_ON_CLOSE );  
    fin.setVisible( b: true );  
}
```



# Listener

Per far funzionare il pulsante è necessario associare un listener (un ascoltatore) in grado di fare un'azione quando il pulsante viene premuto (Action performed)

```
package jbutton_02;
import javax.swing.*.*;
public class JButton_02 {
    public static void main(String[] args) {
        JFrame window = new JFrame( title:"btn counter");
        JPanel panel = new JPanel();
        JButton btn1 = new JButton( text:"Clicca qui");
        btn1.addActionListener(new Counter(panel));
        JButton btn2 = new JButton( text:"Reset");
        btn2.addActionListener(new ResetCounter(panel));
        panel.add( comp:btn1);
        panel.add( comp:btn2);
    }
}
```



Nella main class si crea il frame, il panel e i pulsanti a cui si aggiunge l'**ActionListener** che sarà definito in una classe a parte

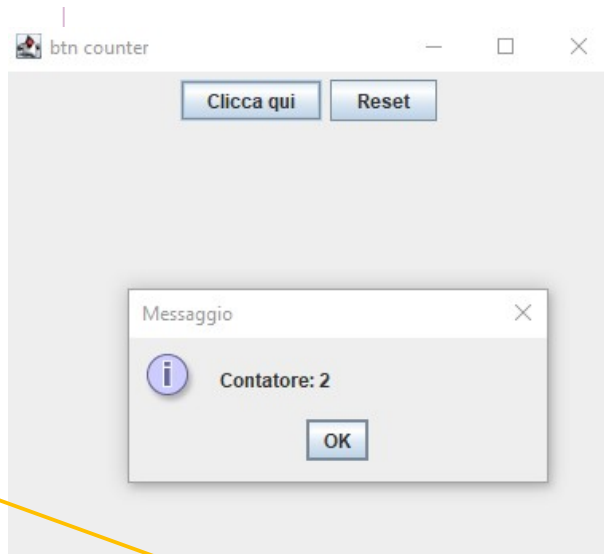
# ActionListener e ActionPerformed

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class Counter implements ActionListener{
    int counter;
    JPanel panel;
    public Counter(JPanel panel){
        counter = 0;
        this.panel = panel;
    }
    public void incCounter(){
        counter++;
    }
    public void actionPerformed(ActionEvent e){
        counter++;
        JOptionPane.showMessageDialog( parentComponent:null, "Contatore: " + counter);
        panel.add(new JLabel("Contatore: " + counter));
    }
}
```

Importante

E' il metodo che viene  
eseguito al click e  
contiene le operazioni  
da fare

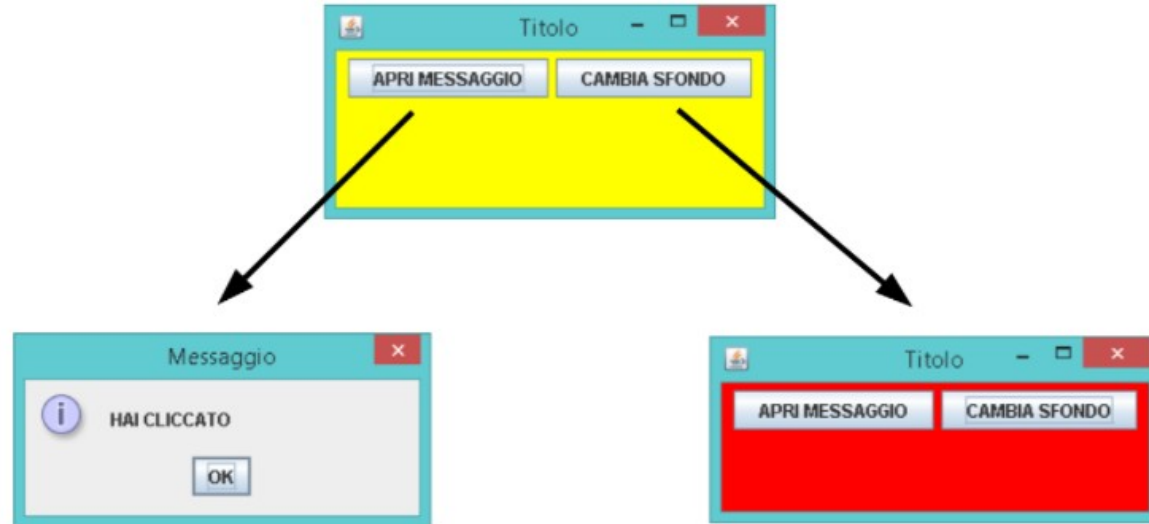
Parametro da  
passare al  
AddActionListener



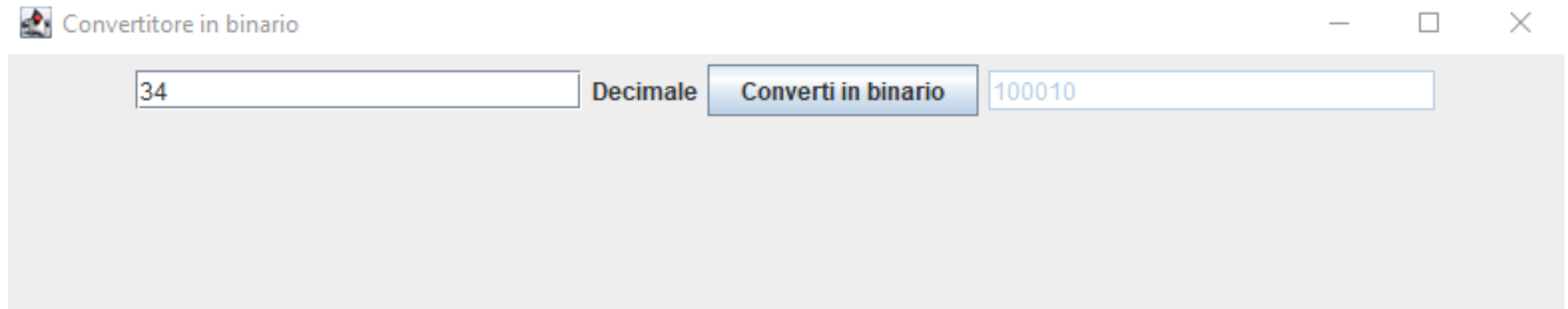


# Esercizio: bottone con cambio sfondo

# Esercizio: bottoni per conteggio click e reset



# Esercizio: bottone conversione in binario



The image shows a graphical user interface window titled "Convertitore in binario". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. Inside the window, there is a text input field containing the number "34". To the right of this field is the label "Decimale". Further right is a blue button with the text "Converti in binario". To the right of the button is another text input field containing the number "100010".

# Esercizio: bottone conversione in binario

```
public static void main(String[] args) {  
    // nuovo oggetto frame  
    JFrame win = new JFrame ( title: "Convertitore in binario");  
    // nuovo oggetto pannello  
    JPanel panel = new JPanel ();  
    // label per pannello  
    JLabel deci = new JLabel( text: "Decimale");  
    // nuovo campo testo per l'input  
    JTextField input = new JTextField( columns: 20);  
    // nuovo campo testo per l'output non editabile  
    JTextField output = new JTextField( columns: 20);  
    output.setEnabled( enabled: false);  
    // nuovo bottone  
    JButton button = new JButton ( text: "Converti in binario");  
  
    // nuovo oggetto listener e aggiunta al bottone  
    Listener n = new Listener (input, output);  
    button.addActionListener( l:n);  
}
```

## Nel main:

- Frame
- Pannello
- Text field editabile (numero decimale)
- Text field non editabile (num.binario)
- Button
- Listener

# Esercizio: bottone conversione in binario

```
public class Listener implements ActionListener {  
    JTextField input;    // due attributi il numero in decimale  
    JTextField output;  // il numero in binario  
  
    public Listener (JTextField input, JTextField output) {  
        this.input=input;        // costruttore con parametri  
        this.output=output;  
    }  
  
    public void actionPerformed (ActionEvent e) {  
        // variabile di supporto per la conversione  
        Integer n=0;            // conversione con la funzione toBinaryString  
        output.setText( Integer.toString(Integer.parseInt(input.getText())) );  
    }  
}
```

Nella classe Listener

- 2 attributi di tipo Text field
- ActionPerformed che trasforma in binario con il metodo toBinaryString di Integer)
- parseInt per convertire in intero da stringa
- getText e setText per prendere e scrivere nel TextField

# Esercizio: bottone per somma di 2 numeri

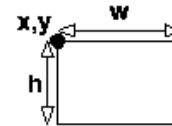
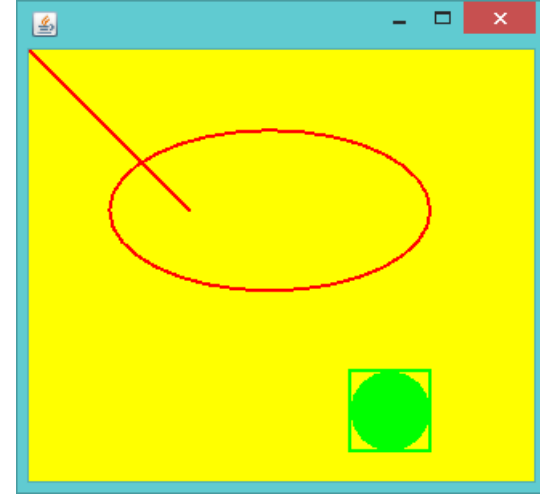


A screenshot of a Windows application window titled "Somma numeri". The window contains a simple addition interface. It features two input text boxes at the top left, containing the numbers "12" and "24". To the right of these boxes is a blue button labeled "Somma". Further to the right is a text box displaying the result "36". The window has a standard Windows title bar with minimize, maximize, and close buttons.

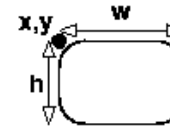
# Disegnare all'interno di un pannello

E' possibile creare grafica personalizzata creando una classe che estende JPanel e ridefinisce il suo metodo **paint()**

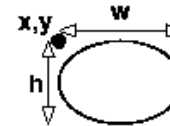
```
class MyPanel extends JPanel {  
  
    public MyPanel() {  
        setBackground(Color.yellow);  
    }  
  
    public void paint(Graphics g) {  
        super.paint(g);  
        Graphics2D g2 = (Graphics2D) g;  
  
        g2.setStroke(new BasicStroke(2));  
        g2.setColor(Color.red);  
        g2.drawLine(0, 0, 100, 100);  
        g2.drawOval(50, 50, 200, 100);  
        g2.setColor(Color.green);  
        g2.fillOval(200, 200, 50, 50);  
        g2.drawRect(200, 200, 50, 50);  
    }  
}
```



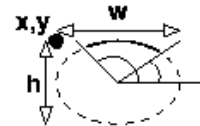
Rect



RoundRect



Oval



Arc

# Graphics

- Ogni componente grafico in Swing possiede un metodo `paint` che si occupa di "disegnare l'oggetto stesso" nella finestra
- Un `JPanel` è un'area "vuota". Ridefinendo ***paint*** è possibile disegnare all'interno di quest'area.
- *paint* riceve come argomento un oggetto di tipo *Graphics*.
- L'oggetto *Graphics* consente di disegnare all'interno dell'area di schermo occupata dal componente.
- *Graphics* ha metodi per tracciare figure vuote (`drawRect`, `drawOval`, ...) o piene (`fillRect`, `fillOval`, ...), immagini, testo, rette, archi, ecc.
- Le coordinate sono relative all'angolo in alto a sinistra del componente ( non dell'intera finestra o del monitor).
- E' possibile cambiare il colore del tratto e lo spessore del tratto (Stroke)



# Graphics

I disegni nel pannello si fanno chiamando metodi della classe Graphics sull'oggetto. Graphics2D è una sottoclasse di Graphics.

La classe Graphics contiene funzionalità grafiche più semplici mentre la classe Graphics2D contiene funzionalità più avanzate

Per usare le funzionalità avanzate si deve fare una conversione esplicita (cast)

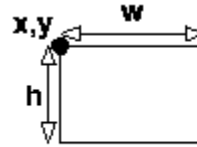
**Graphics2D g2 = (Graphics2D)g;**

## Attributi:

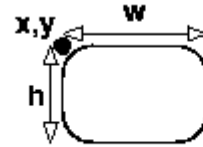
COLOR = colore

STROKE = stile del tratto

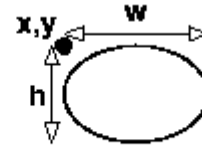
FONT = per le stringhe



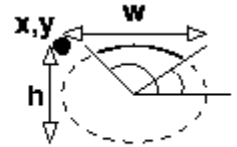
Rect



RoundRect



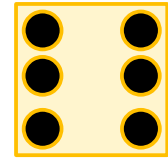
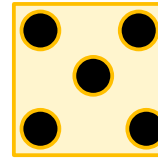
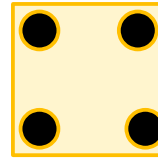
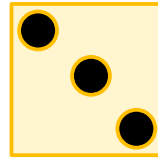
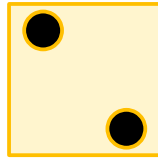
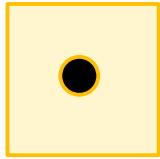
Oval



Arc

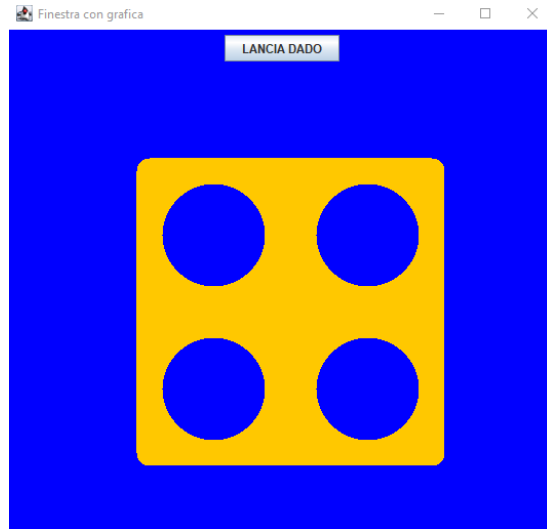
# Esercizio: disegna un dado

- Disegnare la faccia di un dado



# Esercizio: Lancio del dado

Quando viene premuto il bottone viene il disegno di un dado con il numero che è uscito



## Nel main:

- JFrame
- MyPanel (JPanel)
- JButton

## Nella classe MyPanel:

- metodo setNum per prendere in ingresso il random
- disegnare la faccia del dado con Graphics

## Nella classe MyListener:

- due parametri (frame e pannello)
- random
- richiamare il metodo setNum
- ridefinire un nuovo pannello MyPanel (con anche JButton)
- e visualizzare il frame

Suggerimento: per estrarre un numero -----> `val = 6*Math.random()+1;`

# ArrayList

- E' possibile creare un array dinamico di bottoni dove ogni bottone può essere associato (singolarmente) ad un pannello / interfaccia grafica.
- In questo modo quando si crea un ascoltatore si può creare una classe che sarà integrata passando un solo oggetto Array list nel costruttore indipendentemente da quanti oggetti ci sono nella grafica.

```
import java.util.ArrayList; // import the ArrayList class

ArrayList<JButton> cars = new ArrayList<JButton>(); // Create an ArrayList object
```

# Combo Box

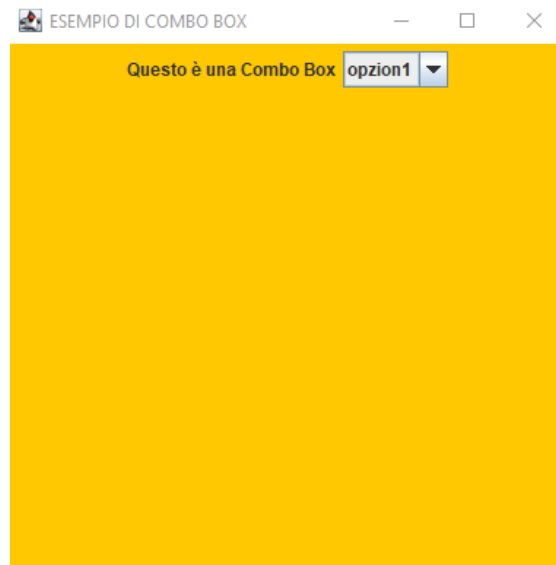
```
JPanel pannello = new JPanel();
pannello.setBackground( bg: Color.orange );
// AGGIUNGO AL PANEL UNA LABEL
pannello.add( new JLabel( text: "Questo è una Combo Box" ) );
// CREO UN VETTORE DI STRINGHE CON LE OPZIONI DELLA COMBO BOX
String[] opzio = { "opzion1", "opzion2", "opzion3", "opzion4", "opzion5" };

JComboBox combo = new JComboBox( items: opzio );
pannello.add( comp: combo );
fin.add( comp: pannello );                                // AGGIUNGO IL PANEL AL FRAME
```

**Per ottenere l'elemento selezionato, è disponibile il metodo:**

`getSelectedItem();`

Associare la combo box ad un listener per ottenere l'elemento selezionato e definire le operazioni successive

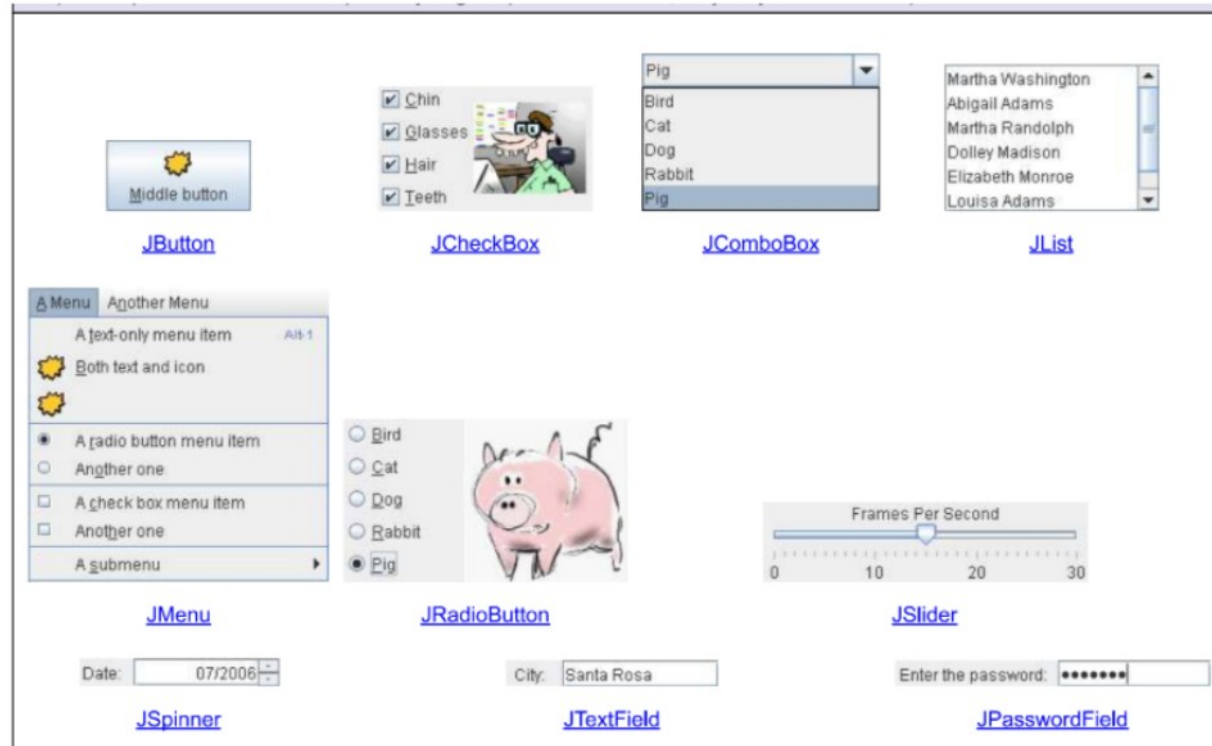


# Esercizi

Crea una finestra con un campo che abbia una tendina (JCombo BOX) e un tasto (Visualizza). Nella tendina puoi scegliere Cerchio, Quadrato, Rettangolo e Rombo. La figura cambia quando clicchi. Il disegno è con Graphics2D.

<https://docs.oracle.com/javase/tutorial/uiswing/components/combobox.html>

# Altri componenti



# Metodi per posizionare senza layout

Togliere il layout di default del pannello:

**pannello.setLayout(null);**

Definire la posizione di ogni elemento nel pannello:

**elemento.setBounds(x,y,width,height);**

```
JPanel pannello = new JPanel();  
pannello.setLayout( mgr:null );  
pannello.setBackground( bg:Color.yellow );  
JLabel titolo1=new JLabel( text:"titolo" );  
titolo1.setBounds( x:50, y:10, width:400, height:30 );  
pannello.add( comp:titolo1 );
```



# Modificare il font di una JLabel

Definire il font:

**Font font1=new Font(name, style, size);**

(esempio: Font font1=new Font("Comic Sans", Font.PLAIN, 15);

Settarlo nella label:

**label.setFont(font1);**

```
JLabel titolo2=new JLabel( text:"Label");  
titolo2.setBounds( x:50, y:75, width:200, height:30);  
Font font1=new Font( name:"Comic Sans", style:Font.PLAIN, size:15);  
titolo2.setFont( font:font1);  
pannello.add( comp:titolo2 );
```