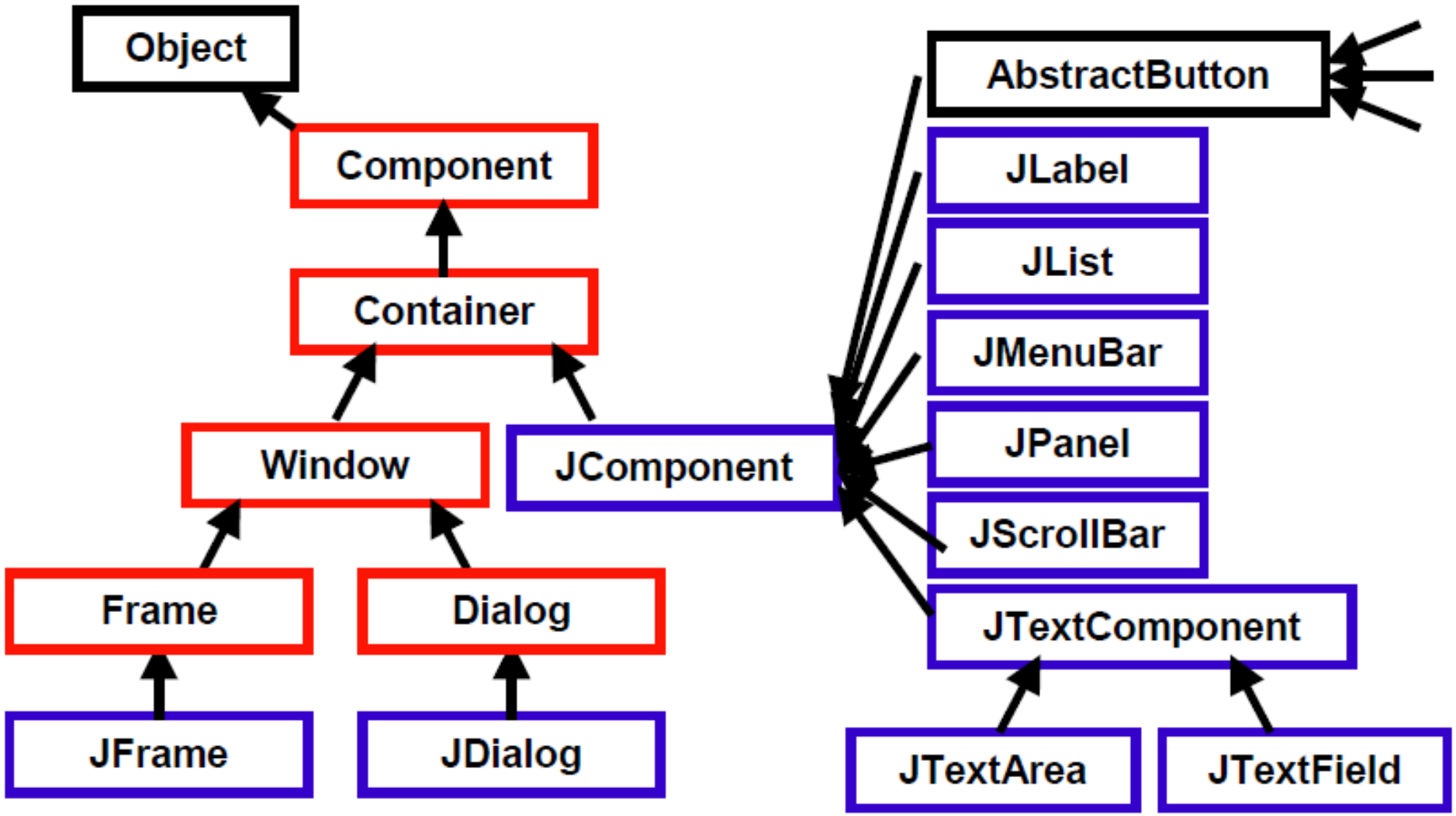
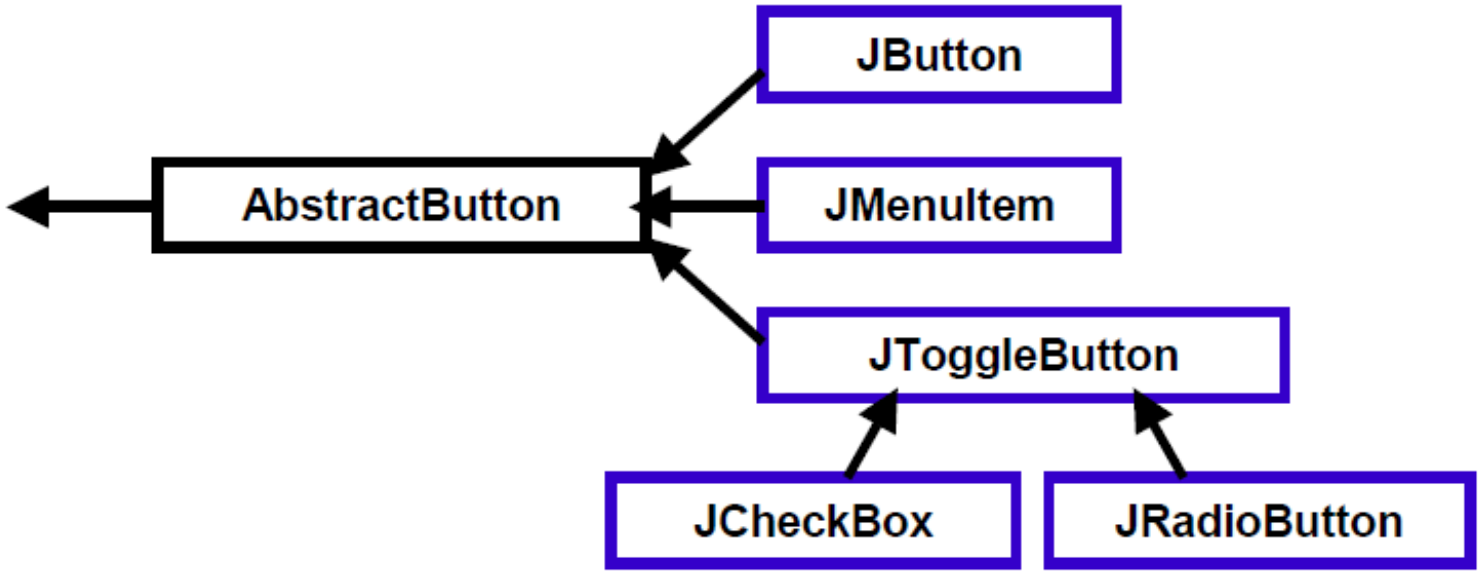
Disegnare su un JPanel

Swing

SWING: GERARCHIA DI CLASSI



SWING: GERARCHIA DI CLASSI



Container: tutti i componenti principali sono contenitori, destinati a contenere altri componenti
Window: le finestre sono casi particolari di contenirori e si distinguono in frame e finestre di dialogo
JFrame: componente finestra principale: ha un aspetto grafico, una cornice ridimensionabile e un titolo
JComponent: è il generico componente grafico
JPanel: il pannello, un componente destinato a contenere altri componenti grafici per organizzarli

Disegnare su un Pannello

Per disegnare su un pannello occorre:

1. definire una propria classe (**MioPannello**) che estenda il **JPanel** originale
2. ridefinire, in tal classe, il metodo ***paintComponent()***, che è il metodo (ereditato da **JComponent**) che si occupa di disegnare il componente

ATTENZIONE: il nuovo ***paintComponent()*** da noi definito deve sempre richiamare il metodo paintComponent() originale, tramite super

```
public class MioPannello extends JPanel {

        // nessun costruttore, va bene il quello di default

        public void paintComponent(Graphics g){
                super.paintComponent(g);
                ...
                // qui aggiungeremo le nostre istruzioni di disegno….
                //g è un oggetto gestito dal sistema a cui ci si
                // rivolge per disegnare
        } //func
} //class
```

Disegnare una figura

Il pannello personalizzato con il disegno:

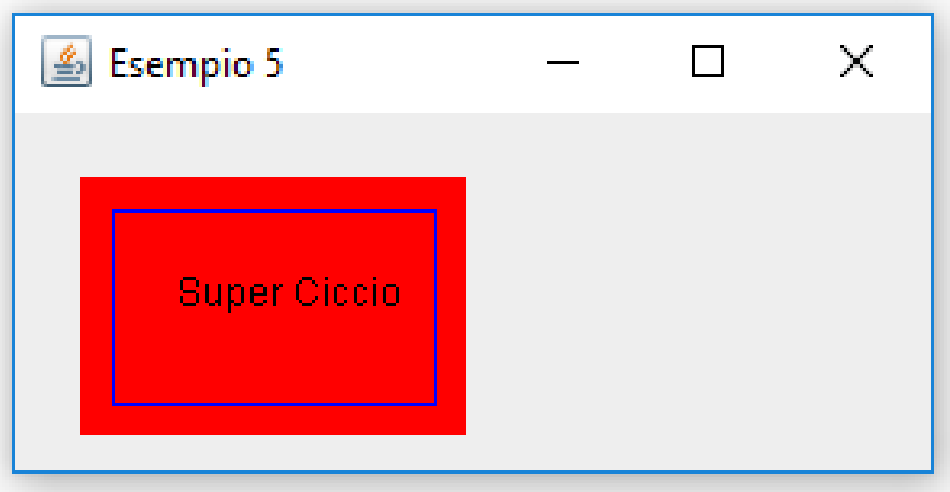
```
public class MioPannello extends JPanel {
      public void paintComponent(Graphics g){
            super.paintComponent(g);
            g.setColor(Color.red);
            // white, gray, red, green, yellow, pink, etc.
            g.fillRect(20,20, 120,80);
            g.setColor(Color.blue);
            g.drawRect(30,30, 100,60);
            g.setColor(Color.black);
            g.drawString("Super Ciccio",50,60);
      } //func
} //class
```

Il main che lo crea e lo inserisce nel frame

```
public class EsempioSwing_n_5{
      public static void main(String[] v){

            MioJFrame f = new MioJFrame("Esempio 5");
            // potremmo anche usare un JFrame standard…
            //cioe scrivere Jframe f = new Jframe();

            Container c = f.getContentPane();
            MioPannello panel = new MioPannello();
            c.add(panel);
            f.setVisible(true);;
      } //main
} //class
```



E se volessimo cambiare FONT?

1. si crea un oggetto Font appropriato (passando al metodo costruttore nome_del_font, stile e dimensione in punti (come word)- stili possibili: Font.PLAIN, Font.ITALIC);

```
Font f1 = new Font("Times", Font.BOLD, 20);
```

2. lo si imposta come font predefinito usando il metodo setFont()

```
g.setFont(f1);
```

Conoscere le proprietà del font corrente

Il font corrente si recupera con getFont()

Dato un Font, le sue proprietà si recuperano con ***getName(), getStyle(), getSize()*** e si verificano con i predicati ***isPlain(), isBold(), isItalic()*** (per sapere se il font è normale, grassetto o corsivo)

```
Font f1 = g.getFont();
int size = f1.getSize();
int style = f1.getStyle();
String name = f1.getName();
```

Esercizio: Interfacci grafica con linee parallele e stringe 1/2

```
public class ImmaginePannello extends JPanel {
        public void paintComponent(Graphics g){
                super.paintComponent(g);
                g.setColor(Color.red); g.drawString("Esercizio",60,40);
                g.drawLine(60, 160, 400, 100);

                //linea blue
                g.setColor(Color.blue); g.drawLine(60, 180, 400, 120);
                //linea nera
                g.setColor(Color.black); g.drawLine(60, 200, 400, 140);
                //linea gialla
                g.setColor(Color.yellow); g.drawLine(60, 220, 400, 160);

                //colore settato di nuovo a blu per la stringa
                g.setColor(Color.blue); g.drawString("font di default",60,250);
                Font f1 = new Font("Times", Font.BOLD, 20);
                g.setFont(f1);
                g.drawString("Times New Roman bold e size: 20",60,280);
        } //func
} //class
```

Esercizio: Interfacci grafica con linee parallele e stringe 2/2

```
public class RigheParallele {
      public static void main(String[] v){
            JFrame f = new JFrame("Disegno e Stringhe");
            Container c = f.getContentPane();
            ImmaginePannello p = new ImmaginePannello();
            c.add(p);
            f.setBounds(100,100,515,440);
            f.setVisible(true);
      } //main
} //class
```

Disegnare su un Panel

Disegnare una funzione: f(x)

Grafico di f(x)

Per disegnare il grafico di una funzione occorre:

1. creare un'apposita classe FunctionPanel che estenda JPanel, ridefinendo il metodo ***paintComponent()*** come appropriato, ad esempio:
   - sfondo bianco, cornice nera;
   - assi cartesiani rossi, con estremi indicati;
   - funzione disegnata in blu;

2. creare, nel main, un oggetto di tipo FunctionPanel

```
import java.awt.*;
import javax.swing.*;
public class EsercizioSwing_n_6 {
        public static void main(String[] v){
                JFrame f = new JFrame("Grafico f(x)");
                Container c = f.getContentPane();
                PannelloDellaFunzione p = new PannelloDellaFunzione();
                c.add(p);
                f.setBounds(100,100,500,400);
                f.setVisible(true);
        } //main
} //class
```

Definizione del pannello

```
class PannelloDellaFunzione extends JPanel {

        //gli intervalli in cui vogliamo graficare
        int xMin=-7, xMax=7, yMin=-1, yMax=1;

        // grandezza della finestra in cui graficare la funzione
        //corrispondono alla grandezza del Jframe
        int larghezza=500, altezza=400; //meglio se altezza e'365
        float fattoreScalaX, fattoreScalaY;


        public void paintComponent(Graphics g){
                super.paintComponent(g); // va fatto sempre
                setBackground(Color.white); //fondo bianco

                // dobbiamo fare le proporzioni tra
                // l'intervallo di valori della finestra
                // ([500*400]pixel) e l'intervallo da graficare [14*2] valori
                fattoreScalaX = larghezza/((float)xMax-xMin);
                fattoreScalaY = altezza/((float)yMax-yMin);
```

```
// incorniciamo il grafico con una cornice in nero
g.setColor(Color.black);
g.drawRect(0,0,larghezza-1,altezza-1);

// disegna degli assi cartesiani in rosso
g.setColor(Color.red);
g.drawLine(0,altezza/2, larghezza-1,altezza/2); //asse x
g.drawLine(larghezza/2,0,larghezza/2,altezza-1); //asse y

// scrittura valori estremi degli assi
//stampa la stringa -7 a partire dalle coordinate x=5, y=195
g.drawString(""+xMin, 5,altezza/2-5);
//stampa la stringa 7 a partire dalle coordinate x=490, y=195
g.drawString(""+xMax, larghezza-10,altezza/2-5);

//stampa la stringa 1 a partire dalle coordinate x=250+5, y=15
g.drawString(""+yMax, larghezza/2+5,15);

//stampa la stringa -1 a partire dalle coordinate x=250+5, y=395
g.drawString(""+yMin, larghezza/2+5,altezza-5);

//disegna il grafico della funzione in blu
g.setColor(Color.blue);
setPixel(g,xMin,f(xMin)); // stampa del punto iniziale
```

```
            // per ognuno dei pixel della finestra (che vanno da 0 a 500)
            //devo calcolare il corrispondente x in scala
            for (int ix=1; ix<larghezza; ix++){
                        float x = xMin+((float)ix)/fattoreScalaX;
                        setPixel(g,x,f(x));
            } //for
      } //func

      // definizione della funzione, statica, da graficare
      static float f(float x){
            //sin(x) è la funzione (statica!) che decidiamo di graficare:ovviamente
            //potrebbe essere qualsiasi funzione
            return (float)Math.sin(x);
      } //func

      // questa serve per riportare i valori della funzione sui valori della finestra
      void setPixel(Graphics g, float x, float y){
            if (x<xMin || x>xMax || y<yMin || y>yMax ) return;
            int ix = Math.round((x-xMin)*fattoreScalaX);
            int iy = altezza-Math.round((y-yMin)*fattoreScalaY);
            // disegna in effetti un singolo punto
            g.drawLine(ix,iy,ix,iy);
      } //func
} //class
```

Grafico di f(x) = sin(x)