Posizionamento diretto di un elemento GUI o tramite gestori di layout

Nel posizionamento diretto (dipendente dalle impostazioni grafiche del sistema) non abbiamo bisogno di un gestore di layout (*layout manager*[1]) e dobbiamo indicare questo fatto con l'istruzione:

setLayout (**null**);

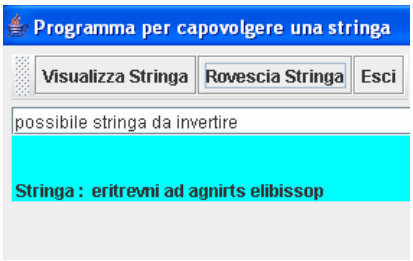
Definiremo poi le dimensioni dell'elemento e lo posizioneremo mediante il metodo **setBounds**() con sintassi:

setBounds(x,y,larghezza,altezza) // x,y dell'angolo sup. sin.

Per default il **gestore di layout** per tutti i **contenitori di tipo pannello** è un istanza della classe ***FlowLayout*** che aggiunge i componenti uno di seguito all'altro da sinistra a destra (nello stesso ordine in cui vengono aggiunti col metodo add): riempita la prima riga si inizia la seconda e così via.

Abbiamo già fatto uso della classe ***GridLayout*** con costruttore **GridLayout (numero righe, numero colonne)** per posizionare i componenti nelle celle di una tabella tutte caratterizzate dalla stessa dimensione. Un altro costruttore permette di impostare anche la spaziatura verticale e quella orizzontale: **GridLayout** (num_righe, num_colonne, spaziatura_verticale, spaziatura_orizzontale)

BorderLayout

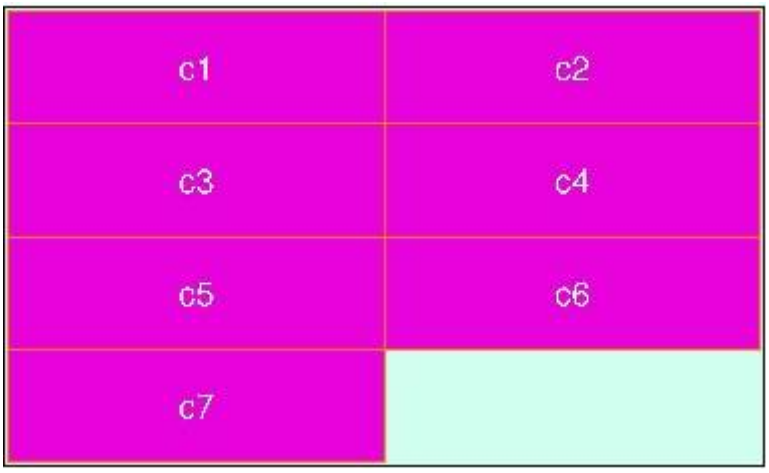


BorderLayout divide lo schermo in cinque regioni: North, East, West, South, Center. È, di default il gestore di layout per i contenitori ad alto livello come i Content Pane.

BorderLayout assicura a North, South, East e West tanto spazio quanto richiedono, lo spazio rimanente viene assegnato a Center.

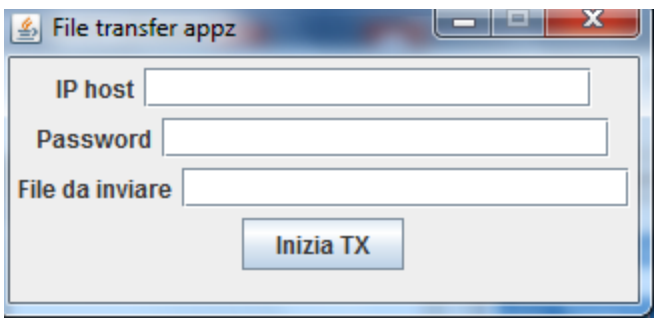
BorderLayout può essere costruito con costruttore **BorderLayout ()** oppure con **BorderLayout (**spaziatura_orizzontale**,** spaziatura_verticale**)** specificano lo spazio orizzontale e verticale tra due componenti.

nb:
nel package *swing* tali regioni vengono definite dalle costanti **BorderLayout.PAGE_START**, **BorderLayout.LINE_START**, **BorderLayout.LINE_END**, **BorderLayout.PAGE_END** e **BorderLayout.CENTER** rispettivamente per indicare in alto, a sinistra, a destra, in basso e centro.
Oltre alle numerose classi che già *awt* introduceva per gestire il posizionamento, *swing* rende disponibile anche la classe **BoxLayout** per una scelta *general purpose* nell'***impilare*** i vari elementi.



1 **Gestori di Layout** http://java.sun.com/docs/books/tutorial/uiswing/layout/visual.html (guida visuale)

```
/*****************************************************************/
/* GESTIONE DI UN LAYOUT SEMPLICE MEDIANTE USO DI SWING ****/
/*****************************************************************/

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

public class SempliceLayout extends JFrame {

        //Creiamo i componenti

        JLabel ipLabel = new JLabel("IP host", SwingConstants.LEFT);              // per default a sinistra
        JLabel passwordLabel = new JLabel("Password", SwingConstants.LEFT);
        JLabel fileDaInviareLabel = new JLabel("File da inviare", SwingConstants.LEFT);

        JTextField ipText = new JTextField(20);                          // impostiamo numero caratteri
        JPasswordField passwordText = new JPasswordField(20);   // impostiamo numero caratteri
        JTextField fileDaInviareText = new JTextField(20);           // impostiamo numero caratteri
        JButton pulsante = new JButton("Inizia TX");

        public SempliceLayout() {                                  // costruttore dell'applicazione

        super("File transfer appz");
        setSize(320, 150);                                                          // dimensioni adatte [*]
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        JPanel pannello = new JPanel();

        // impostiamo le proprietà dei componenti

        ipText.setEditable(true);
        fileDaInviareText.setEditable(true);
        passwordText.setEchoChar('*');

        // usiamo il Layout di default di JPanel di tipo FlowLayout

        pannello.add(ipLabel);
        pannello.add(ipText);
        pannello.add(passwordLabel);
        pannello.add(passwordText);
        pannello.add(fileDaInviareLabel);
        pannello.add(fileDaInviareText);

        pannello.add(pulsante);
        setContentPane(pannello);  // rendiamo il pannello parte del nostro frame
        setResizable(false);                                                    // immodificabili dimensioni [*]
        setVisible(true);       // visualizziamo il tutto
        }

        public static void main(String argv[]) {

        new SempliceLayout();

        }
}
```

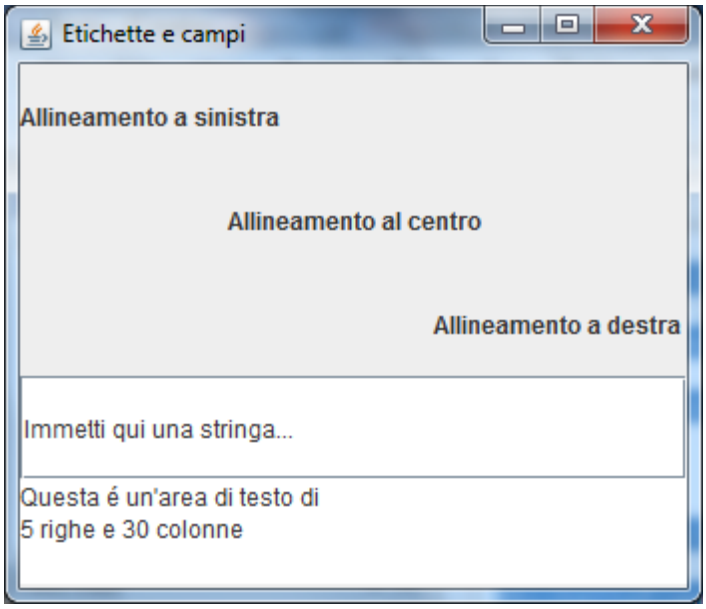


```
/*****************************************************************************/
/* GESTIONE DI UN LAYOUT A GRIGLIA AVANZATA MEDIANTE USO DI SWING ****/
/*****************************************************************************/

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

public class Layout extends JFrame {

//Creiamo i componenti

JLabel ipLabel = new JLabel("IP host", SwingConstants.LEFT);              // per default a sinistra
JLabel passwordLabel = new JLabel("Password", SwingConstants.LEFT);
JLabel fileDaInviareLabel = new JLabel("File da inviare", SwingConstants.LEFT);
JTextField ipText = new JTextField();                        // non ci preoccupiamo di impostare numero caratteri
JPasswordField passwordText = new JPasswordField();
JTextField fileDaInviareText = new JTextField();
JButton pulsante = new JButton("Inizia TX");

// Definiamo un metodo che ci servirà per definire i limiti di layout

void impostaLimite(GridBagConstraints gbc, int gx, int gy, int gw, int gh, int wx, int wy) {
gbc.gridx = gx;
gbc.gridy = gy;
gbc.gridwidth = gw;
gbc.gridheight = gh;
gbc.weightx = wx;
gbc.weighty = wy;
}

public Layout() {                  // costruttore dell'applicazione

super("File transfer appz");
setSize(300, 120);
setDefaultCloseOperation(EXIT_ON_CLOSE);
JPanel pannello = new JPanel();

// impostiamo le proprietà dei componenti

ipText.setEditable(true);
fileDaInviareText.setEditable(true);
passwordText.setEchoChar('*');

//definiamo il gestore di layout

GridBagLayout grigliaAvanzata = new GridBagLayout();    // griglia personalizzata
GridBagConstraints limite = new GridBagConstraints();
pannello.setLayout(grigliaAvanzata);

//definiamo i limiti di ogni componente e lo aggiungiamo al pannello

impostaLimite(limite,0,0,1,1,35,0);                  //etichetta IP host
limite.fill = GridBagConstraints.NONE;
limite.anchor = GridBagConstraints.EAST;
grigliaAvanzata.setConstraints(ipLabel,limite);

pannello.add(ipLabel);

impostaLimite(limite,1,0,1,1,65,100);                  //campo IP host
limite.fill = GridBagConstraints.HORIZONTAL;
grigliaAvanzata.setConstraints(ipText,limite);

pannello.add(ipText);
```

```
impostaLimite(limite,0,1,1,1,0,0);                    //etichetta password
limite.fill = GridBagConstraints.NONE;
limite.anchor = GridBagConstraints.EAST;
grigliaAvanzata.setConstraints(passwordLabel,limite);

pannello.add(passwordLabel);

impostaLimite(limite,1,1,1,1,0,100);                  //campo password
limite.fill = GridBagConstraints.HORIZONTAL;
grigliaAvanzata.setConstraints(passwordText,limite);

pannello.add(passwordText);


impostaLimite(limite,0,2,1,1,0,0);                    //etichetta File da inviare
limite.fill = GridBagConstraints.NONE;
limite.anchor = GridBagConstraints.EAST;
grigliaAvanzata.setConstraints(fileDaInviareLabel,limite);

pannello.add(fileDaInviareLabel);

impostaLimite(limite,1,2,1,1,0,100);                  //campo File da inviare
limite.fill = GridBagConstraints.HORIZONTAL;
grigliaAvanzata.setConstraints(fileDaInviareText,limite);

pannello.add(fileDaInviareText);

impostaLimite(limite,0,3,2,1,0,50);                    // Pulsante
limite.fill = GridBagConstraints.NONE;
limite.anchor = GridBagConstraints.CENTER;
grigliaAvanzata.setConstraints(pulsante,limite);


pannello.add(pulsante);

setContentPane(pannello);          // rendiamo il pannello parte del nostro frame
setVisible(true);     // Visualizziamo il tutto!
}

public static void main(String argv[]) {

Layout nf = new Layout();


}


}
```

GridBagLayout può organizzare interfacce grafiche complesse rendendo possibile dividere il container in una griglia e, a differenza del GridLayout, può disporre i suoi componenti in modo tale che si estendano anche oltre un'unica cella, attraverso la classe **GridBagConstraints**.

```
/*****************************************************************************/
/* SEMPLICI ETICHETTE, CAMPI ED AREE DI TESTO MEDIANTE USO DI SWING  ****/
/*****************************************************************************/

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

public class EtichetteEcampi extends JFrame {

JLabel etichetta1 = new JLabel("Allineamento a sinistra ",SwingConstants.LEFT);
JLabel etichetta2 = new JLabel("Allineamento al centro",SwingConstants.CENTER);
JLabel etichetta3 = new JLabel("Allineamento a destra ", SwingConstants.RIGHT);
JTextField campoDiTesto = new JTextField("Immetti qui una stringa...",30);
JTextArea areaDiTesto = new JTextArea("Questa é un'area di testo di\n5 righe e 30 colonne",5,30);

public EtichetteEcampi() {

        super("Etichette e campi");
        setSize(350, 300);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        JPanel pannello = new JPanel();

        // impostiamo le proprietà dei componenti

        campoDiTesto.setEditable(true);
        areaDiTesto.setLineWrap(true);
        areaDiTesto.setWrapStyleWord(true);

        // ora aggiungiamo i componenti al pannello
        pannello.setLayout(new GridLayout(0,1));
        pannello.add(etichetta1);
        pannello.add(etichetta2);
        pannello.add(etichetta3);

        pannello.add(campoDiTesto);

        pannello.add(areaDiTesto);

        // rendiamo il pannello parte del nostro frame

        setContentPane(pannello);
        setVisible(true); // Visualizziamo il tutto!

        }

        public static void main(String argv[]) {

        EtichetteEcampi ec = new EtichetteEcampi();

        }
}
```

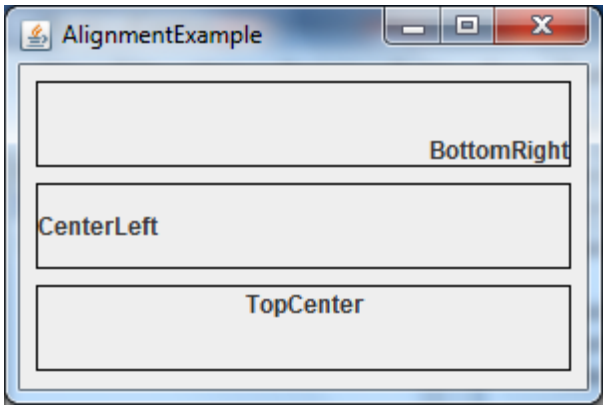


nb: public **GridLayout** (int rows, int cols) crea un layout a griglia dove tutti i componenti sono delle stesse dimensioni. Uno tra numero rige e colonne - ma non entrtambi - può essere **zero**: significa che un qualsiasi numero di oggetti può essere inserito o nell'unica riga o nell'unica colonna

// da http://www.java2s.com/Code/JavaAPI/javax.swing/SwingConstantsRIGHT.htm

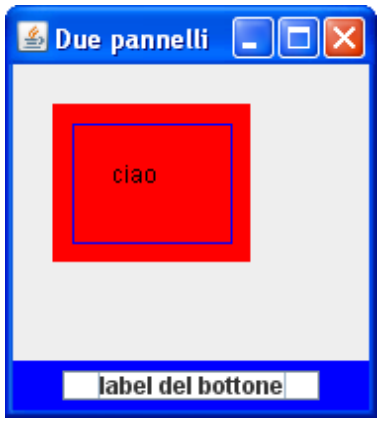
Etichette e bordi

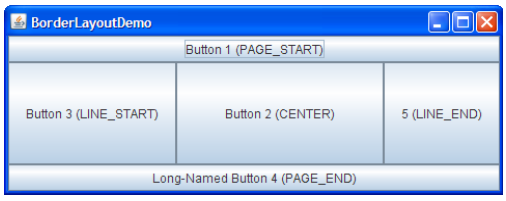
```
import java.awt.Color;
import java.awt.GridLayout;

import javax.swing.BorderFactory;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.SwingConstants;

public class EtichetteBordi {

        private JLabel label1, label2,label3;
        private JFrame frame;
        private  JPanel p;

        public EtichetteBordi(){

            label1 = new JLabel("BottomRight", SwingConstants.RIGHT);
            label2 = new JLabel("CenterLeft", SwingConstants.LEFT);
            label3 = new JLabel("TopCenter", SwingConstants.CENTER);

            label1.setVerticalAlignment(SwingConstants.BOTTOM);
            label2.setVerticalAlignment(SwingConstants.CENTER);
            label3.setVerticalAlignment(SwingConstants.TOP);

            label1.setBorder(BorderFactory.createLineBorder(Color.black));
            label2.setBorder(BorderFactory.createLineBorder(Color.black));
            label3.setBorder(BorderFactory.createLineBorder(Color.black));

            frame = new JFrame("AlignmentExample");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            p = new JPanel(new GridLayout(3, 1, 0, 8));  // righe, colonne, pixel tra colonne, pixel tra righe
                                                         // in questa applicazione, con unica colonna,
                                                         // è indifferente hgap che può valere anche 0
            p.add(label1);
            p.add(label2);
            p.add(label3);
            p.setBorder(BorderFactory.createEmptyBorder(8, 8, 8, 8));  // top, left, bottom, right
            frame.setContentPane(p);
            frame.setSize(300, 200);
            frame.setVisible(true);
        }

        public static void main(String[] args) {
            new EtichetteBordi();
        }
}
```



nb: public **GridLayout** (int rows, int cols, int hgap, int vgap) crea un layout a griglia dove tutti i componenti sono delle stesse dimensioni e rende possibile impostare il numero di pixel tra le colonne come per il margine destro e sinistro dai bordi (horizontal gap) ed il numero di pixel tra le righe come per i margini in alto e basso dai bordi (vertical gap).

```
/**
 * Gui.java
 * Gui application
 * @author 3AI
 * @version 1.00 2009/4/4
 */
import java.awt.*;
import javax.swing.*;

class MioPanel extends JPanel{
        public MioPanel(){
                        this.setPreferredSize(new Dimension(180, 150));  // per reimpostare dinamicamente
                       }
        public void paintComponent(Graphics g){
                       super.paintComponent(g);
                       g.setColor(Color.red);
                       g.fillRect(20,20, 100,80);
                       g.setColor(Color.blue);
                       g.drawRect(30,30, 80,60);
                       g.setColor(Color.black);
                       g.drawString("ciao",50,60);
                       }

}
class Bottone extends JPanel{

        private JButton b;
        public Bottone(){

                b = new JButton("label del bottone");
                b.setBackground(Color.white);
                b. setPreferredSize(new Dimension(130, 15));
                this.add(b);  // oggetto corrente
        }
}
public class Gui {

        private JFrame f;
        private Container c;
        private MioPanel p;   // pannelli personalizzati
        private Bottone p2;

        public Gui(){
           f = new JFrame("Due pannelli"); // crea un nuovo JFrame Inizialmente invisibile con titolo
           c = f.getContentPane();  // recupera il "muro grezzo"
           p = new MioPanel();   // crea pannello personalizzato adattabile alle dimensioni della finestra
           p2 = new Bottone();   // crea altro pannello personalizzato adattabile alle dimensioni di bottone e finestra
           p2.setBackground(Color.blue);
           c.setLayout(new BorderLayout());  // scelta di Layout che distingue tra alto, basso, destra, sinistra e centro
           c.add(p,BorderLayout.CENTER);  // al centro il pannello con disegno
           c.add(p2,BorderLayout.PAGE_END);  // in basso il bottone (per retrocompatibiltà anche SOUTH)

           f.pack();                    // ottimizza le dimensioni della finestra
           f.setLocation(400,150);
           f.setVisible(true);          // mostra il JFrame
           f.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        }
        public static void main(String[] v){
           Gui gui = new Gui();
        }
}
```



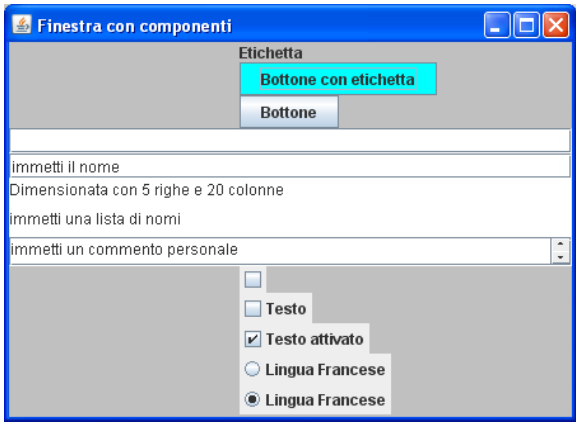


// Lo stesso esempio di Applicazione con uso GUI già usato per introdurre componenti GUI
// in un pannello. **Posizionamento tramite gestori di layout: *impilando in verticale***

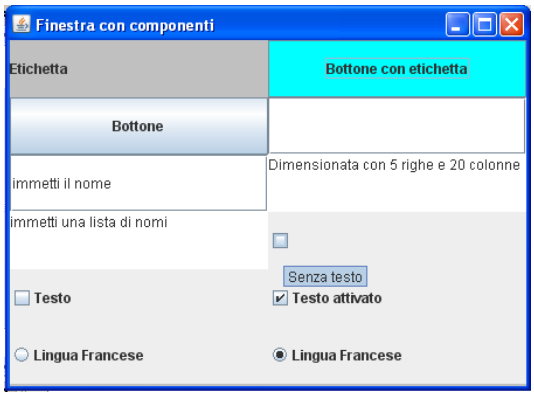
```
import java.awt.*;
import javax.swing.*;
public class Gui2 {

        private  JFrame f ;
        private Container  c;
        private JPanel p;
        private JLabel  l;
        private  JButton b1, b2;
        private  JTextField t1, t2;
        private  JTextArea a1, a2, a3;
        private  JScrollPane sp;                    // con barre a scorrimento inserendo JTextArea in JScrollPane
        private   JCheckBox c1, c2, c3:
        private ButtonGroup g;                      // per creare pulsanti di opzione (uno solo selezionabile) detti Radio Button
        private  JRadioButton rb1, rb2;

        public Gui2 ()    {      // costruttore
             f = new JFrame("Finestra con componenti");                                  // crea frame invisibile
                                                // per impostare le dimensioni e la posizione:
             f.setSize(450,330);                // misure in pixel
             f.setLocation(200,100);            // (0,0) angolo sup. sin.
             f.setResizable(true);                                                       // per ridimensionare con mouse
             c = f.getContentPane();                                                     // recupera il "muro grezzo"
             p = new JPanel();
             p.setBackground (Color.lightGray);    // sfondo colorato

              p.setLayout (new BoxLayout (p, BoxLayout.PAGE_AXIS));   // impila gli elementi in verticale  nel pannello

             l = new JLabel ("Etichetta");

             b1 = new JButton ();
             b2 = new JButton ("Bottone");
             b1.setBackground (Color.cyan);
             b1.setText("Bottone con etichetta");  // imposta il valore dell'etichetta ( deprecato setLabel (testo) )

             t1 = new JTextField (30);
             t2 = new JTextField ("immetti il nome", 20);

              a1 = new JTextArea(5, 20);
              a1.append("Dimensionata con 5 righe e 20 colonne");
              a2 = new JTextArea("immetti una lista di nomi ", 5, 20);
              a3 = new JTextArea("immetti un commento personale ", 2, 15);
              sp = new JScrollPane(a3);                                          // per vedere barre a scorrimento in JTextArea

              c1 = new JCheckBox();
              c2 = new JCheckBox("Testo");
              c3 = new JCheckBox("Testo attivato", true);

             c1.setToolTipText("Senza testo");  // per visualizzare, al passaggio del mouse, testo esplicativo

             // Per creare pulsanti di opzione (uno solo selezionabile) detti Radio Button
              g = new ButtonGroup();  // per gestire selezione
              rb1 = new JRadioButton("Lingua Francese", false);
              rb2 = new JRadioButton("Lingua Francese", true);
              g.add(rb1);
              g.add(rb2);

              p.add(l);   // aggiunge al pannello un'etichetta con testo
              p.add(b1);  // aggiunge al pannello un bottone colorato con etichetta
              p.add(b2);  // aggiunge al pannello un bottone con testo
              p.add(t1);  // aggiunge al pannello un campo di testo con ampiezza specificata
              p.add(t2);   // aggiunge al pannello un campo di testo con inizializzazione  ed ampiezza specificata
             p.add(a1);   // aggiunge al pannello un' area di testo con Righe e Colonne specificate
             p.add(a2);   // aggiunge al pannello un'area di testo con inizializzazione  e Righe e Colonne specificate
             p.add(sp);  // aggiunge al pannello un Pannello a scorrimento che contiene
                         // un'area di testo con inizializzazione e Righe e Colonne ridotte  per visualizzare barre a scorrimento
              p.add(c1);   // aggiunge al pannello una casella di controllo vuota e disattivata
```

```
            p.add(c2);   // aggiunge al pannello una casella di controllo con stringa
            p.add(c3);   // aggiunge al pannello una casella di controllo con stringa ed attivata
            p.add(rb1);  // aggiunge al pannello i Radio Button
            p.add(rb2);

            c.add(p);  // aggiunge il pannello
            f.setVisible(true);                 // mostra il frame (dimensioni 400x400)
            f.setDefaultCloseOperation ( JFrame.EXIT_ON_CLOSE);
           }

        public static void main(String [] args) {
            Gui2 o = new Gui2();
            }
}
```

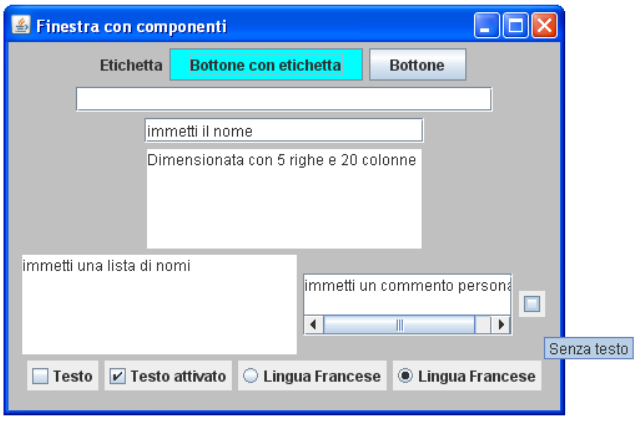
nb: per "impilare" in orizzontale BoxLayout (nomeContenitore, **BoxLayout.LINE_AXIS**)



Impilando gli elementi **in verticale nel pannello**



Impostando 6 Righe e 2 colonne
(con solo 12 componenti)



Default: un componente dopo l'altro

Applet e disposizione di componenti in altra finestra

Lo stesso modo di introdurre componenti GUI si può realizzare con **Applet** o piuttosto **JApplet** se si usano JComponent. Vediamo un esempio che crea una finestra diversa da quella del browser (confrontare con l'applicazione Gui):

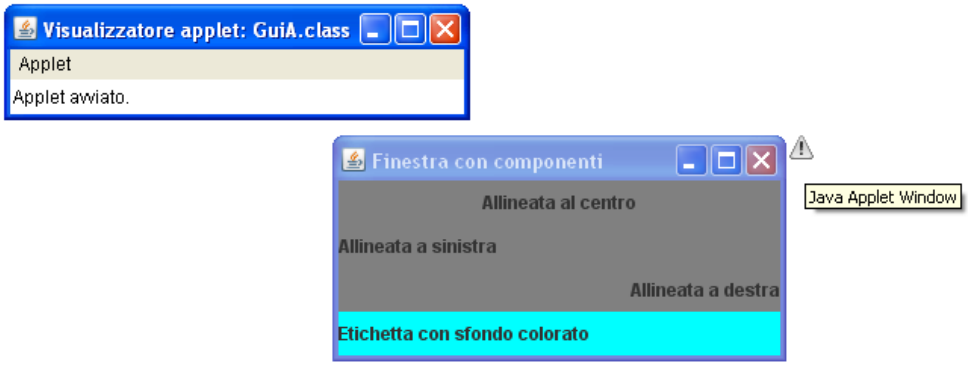
// esempio di **JApplet** con uso dei componenti di una GUI (chiusura finestra browser per uscire)


```
import java.awt.*;

import javax.swing.*;

import java.applet.*;

public class GuiA extends JApplet {

    public void init()   {

        JFrame f = new JFrame("Finestra con componenti"); // crea frame invisibile

        f.setSize(300,150);                 // misure in pixel per impostare le dimensioni
        f.setLocation(200,100);             // e la posizione con (0,0) angolo sup. sin.

        f.setResizable(true); // per ridimensionare con mouse

        Container  c = f.getContentPane();     // recupera il "muro grezzo"

        JPanel p = new JPanel();
        p.setBackground (Color.gray);    // sfondo colorato

        p.setLayout(new GridLayout(4,4)); // per evidenziare allineamento

        JLabel l1 = new JLabel ("Etichetta con sfondo colorato");
        JLabel l2 = new JLabel ("Allineata al centro", JLabel.CENTER);
        JLabel l3 = new JLabel ("Allineata a sinistra", JLabel.LEFT);
        JLabel l4 = new JLabel ("Allineata a destra", JLabel.RIGHT);

        l1.setBackground (Color.cyan);
        l1.setOpaque(true);                      // vedere sfondo su sfondo

        p.add(l2);  // aggiunge al pannello un'etichetta allineata al centro
        p.add(l3);  // aggiunge al pannello un'etichetta allineata a sinistra
        p.add(l4);  // aggiunge al pannello un'etichetta allineata a destra
        p.add(l1);  // aggiunge al pannello un'etichetta con sfondo colorato

        c.add(p);  // aggiunge al frame il pannello colorato

        f.setVisible(true);    // mostra il frame (dimensioni 300x150)

        //  NB: si noti come per ragioni di sicurezza sia proibito l'uso di
                                //  f.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
    }
}
```

Nel codice HTML: <**object** code ="GuiA.class" width= "300" **height= "0"**></object>

le **dimensioni obbligatorie** pur se nulle, qui sono scelte in modo da permettere di leggere completamente, eseguendo in ambiente **JCreator**, il titolo della cornice del visualizzatore della *Java Applet Window* con messaggio di Applet avviato o non avviato.

NB: Un' applet dunque può aprire un'altra finestra *ma compare automaticamente un avviso* anche solo come *piccola icona di allerta* in alto a sinistra della nuova finestra che ricorda come la ***Java Applet Window*** non sia mai completamente invisibile.

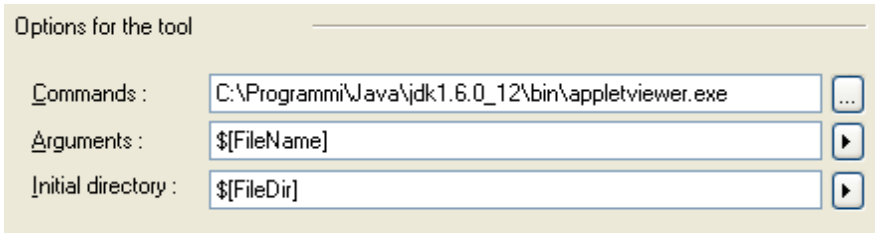


nb: per eseguire la **pagina html** con **opzione Run** in JCreator, salvare tale pagina web nella stessa cartella del bytecode (**classes**). Si noti che,scegliendo come progetto **Basic Java Applet** si crea nella sottocartella **classes** la seguente pagina


```
<html>
    <head> </head>
    <body bgcolor="000000">
        <center><applet code  = "NomeClasse.class"
                         width  = "500"
                         height = "300"        >
            </applet>
        </center>
    </body>
</html>
```

con uso di **attributo sconsigliato** bgcolor e **tag obsoleto** <applet></applet>

nb: per eseguire la pagina html con programma **appletviewer**, inserito come ***tools*** in JCreator, impostare come opzioni:



Esercizio:

Creare un'applet che disegni il cruscotto di un'automobile in una finestra separata. Nel cruscotto devono apparire

- 4 bottoni con testo: Freccia Sx, Freccia Dx, Freno Manuale, Avvia Motore
- un'etichetta con testo Luci con associate 3 caselle opzione (esclusiva) con testo posizioni, anabbaglianti, abbaglianti , tutte non attivate
- un'etichetta carburante
- un JTextField largo 30 pixel

Soluzione:

// esempio di **JApplet** con uso contenitori e componenti GUI del package swing

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

public class Cruscotto extends JApplet  {

    public void init() {

        JFrame f = new JFrame ("Cruscotto di un auto");
        f.setSize (570, 150);
        f.setLocation (100,100);
        f.setResizable(true);    // con mouse ridimensionabile

        Container  c = f.getContentPane();  // recupera il "muro grezzo"

        ButtonGroup luci = new ButtonGroup();
        JRadioButton rb1 = new JRadioButton("posizioni", false);
        JRadioButton rb2 = new JRadioButton("anabbaglianti", false);
        JRadioButton rb3 = new JRadioButton("abbaglianti", false);
        luci.add(rb1);
        luci.add(rb2);

        JButton FrecciaSx = new JButton ("Freccia Sx");
        JButton FrecciaDx = new JButton ("Freccia Dx");
        JButton FrenoManuale = new JButton ("Freno Manuale");
        JButton AvviaMotore = new JButton ("Avvia Motore");

        JTextField carburante = new JTextField (30);

        c.setLayout (new GridLayout (3,4,10,10));
        c.add(FrecciaSx);
        c.add(FrecciaDx);
        c.add(AvviaMotore);
        c.add(FrenoManuale);
        c.add(new Label("Luci"));
        c.add(rb1);
        c.add(rb2);
        c.add(rb3);
        c.add(new Label("Carburante"));
        c.add(carburante);
        f.setVisible(true);
      }
}
```

Nel codice HTML <**object** code ="Cruscotto.class" **width= "0" height= "0"**></object>