I Layout Manager di java

Prof. Francesco Accarino
IIS Altiero Spinelli via Leopardi 132
Sesto san Giovanni

Creazione di interfacce complesse con i layout manager

La posizione di un componente aggiunto a un container dipende essenzialmente:

- dall'oggetto che è associato al container, detto "layout manager". In ogni container infatti, esiste un layout manager associato di default.
- Un layout manager è un'istanza di una classe che implementa l'interfaccia LayoutManager.

layout manager

- Le applicazioni grafiche si basano sempre su di un “top level container”, ovvero un container di primo livello. Infatti, avete mai visto un’applicazione dove ci sono dei bottoni, ma non c’è una finestra che li contiene?
- Questo significa **che in ogni applicazione Java** con interfaccia grafica, è necessario quantomeno istanziare un top level container, di solito un Frame.

- Un caso speciale è rappresentato dalle applet,

layout manager

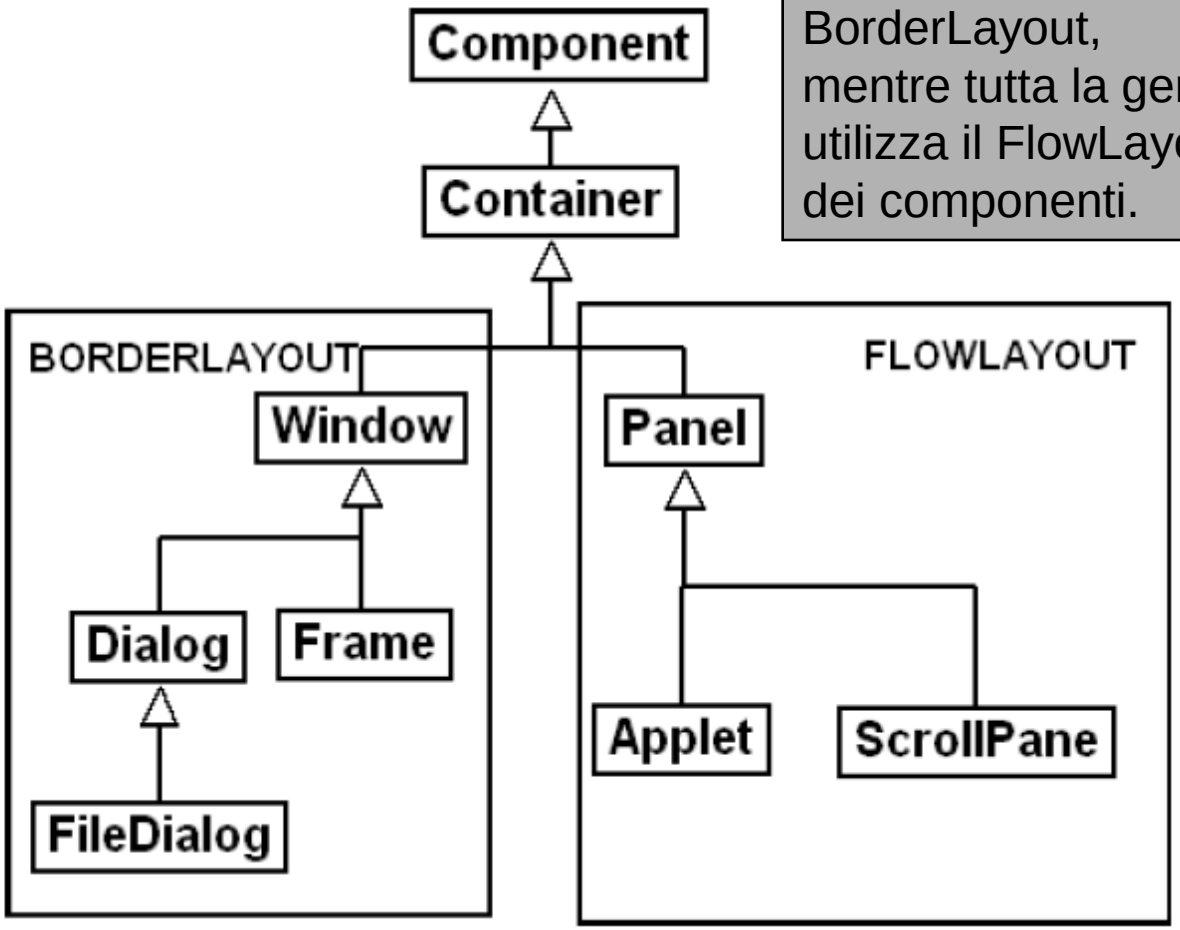
Esistono decine di implementazioni di LayoutManager, ma tutto sommato le più importanti sono solo

cinque:

- FlowLayout
- BorderLayout
- GridLayout
- CardLayout
- GridBagLayout

layout manager

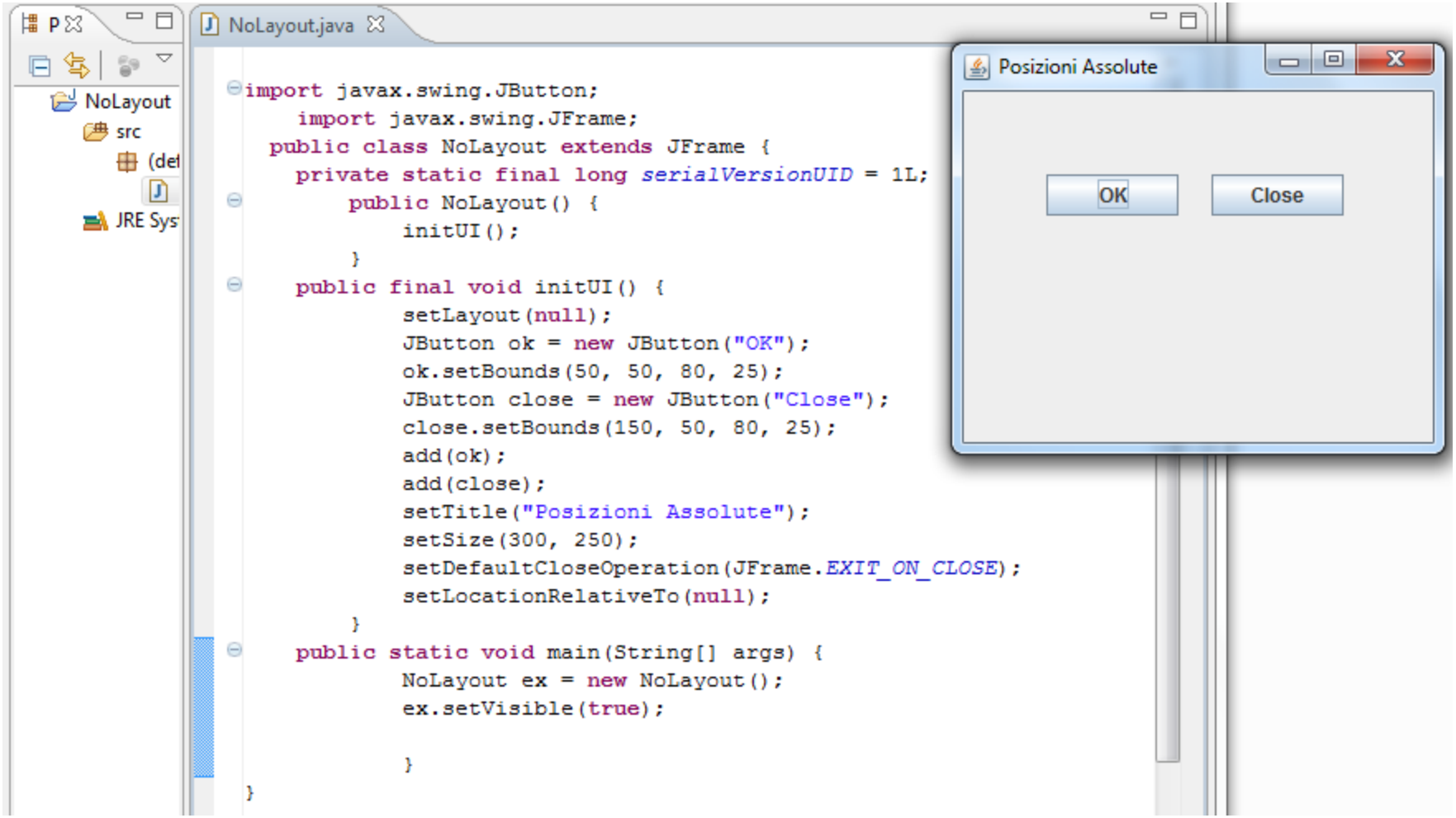
tutte le sottoclassi di Window (quindi anche Frame), anno associato per default il BorderLayout,
mentre tutta la gerarchia di Panel utilizza il FlowLayout per il posizionamento dei componenti.



layout manager

- In realtà è anche possibile non utilizzare layout manager per gestire interfacce grafiche.
- Tale tecnica però comprometterebbe la consistenza e la portabilità della GUI stessa.
- Per esempio si può annullare il layout di un container (ad esempio un Frame) con l'istruzione setLayout(null)
- Poi si usano i metodi setLocation(), setBounds() e setSize() per gestire il posizionamento dei componenti.

Posizioni Assolute senza Layout

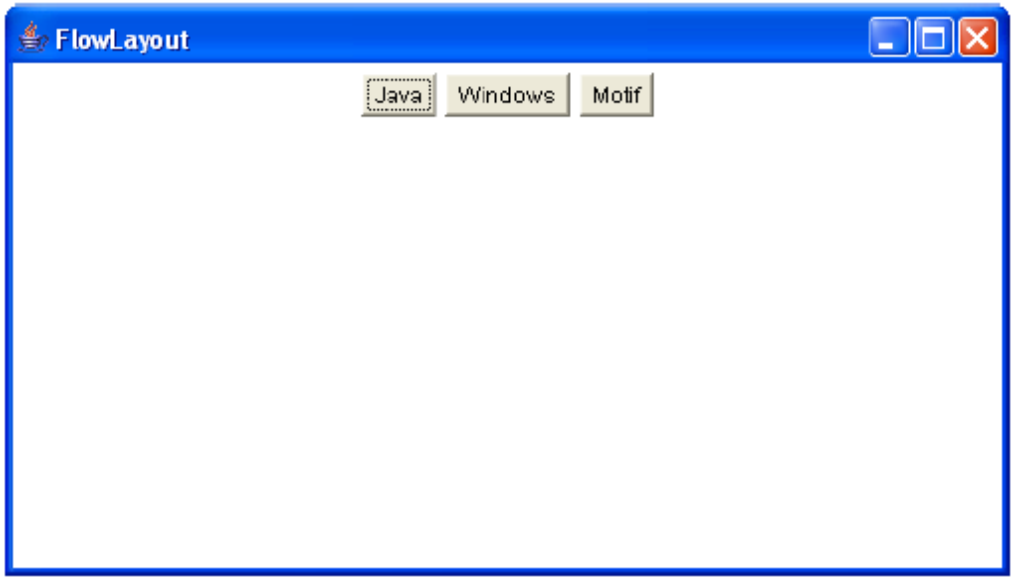


Il FlowLayout

- Il FlowLayout è il layout manager di default di Panel e JPanel.
- FlowLayout dispone i componenti aggiunti in un flusso ordinato che va da sinistra a destra con un allineamento centrato verso l'alto.
- Per esempio il codice seguente (da inserire all'interno di un metodo main()):

```
JFrame f = new JFrame("FlowLayout");
JPanel p = new JPanel();
JButton button1 = new JButton("Java");
JButton button2 = new JButton("Windows");
JButton button3 = new JButton("Motif");
p.add(button1);
p.add(button2);
p.add(button3);
f.add(p);
f.pack();
f.setVisible(true);
```

Effetti di ridimensionamento sulla finestra



FlowLayout pannello di default per Jpanel

FlowLayout.java

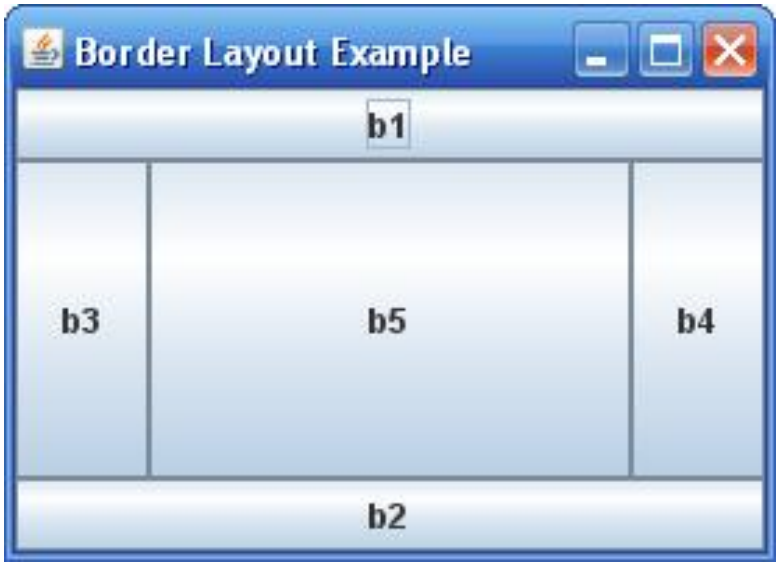
```
import javax.swing.*;
public class FlowLayout extends JFrame {
    private static final long serialVersionUID = 1L;
    public FlowLayout() {
        initUI();
    }
    public final void initUI() {
        JPanel panel = new JPanel();
        JTextArea area = new JTextArea("Questa è una casella di testo");
        area.setSize(100, 100);
        JButton button = new JButton("Bottone");
        panel.add(button);
        JTree tree = new JTree();
        panel.add(tree);
        panel.add(area);
        add(panel);
        pack();
        setTitle("Esempio di FlowLayout");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setVisible(true);
    }
    public static void main(String[] args) {
                new FlowLayout();
    }
}
```

Il BorderLayout

- Il BorderLayout è il layout manager di default per i Frame, il top level container pereccellenza.
- I componenti sono disposti solamente in cinque posizioni specifiche che si ridimensionano automaticamente:
- NORTH, SOUTH che si ridimensionano orizzontalmente
- EAST, WEST che si ridimensionano verticalmente
- CENTER che si ridimensiona orizzontalmente e verticalmente

Esempio

```
import java.awt.*;
import javax.swing.*;
public class BorderExample {
private JFrame f;
private JButton b[]={new JButton("b1"),new JButton("b2"),new JButton("b3"),
new JButton("b4"), new JButton("b5")};
public BorderExample() {
f = new JFrame("Border Layout Example");
}
public void setup() {
f.add(b[0], BorderLayout.NORTH);
f.add(b[1], BorderLayout.SOUTH);
f.add(b[2], BorderLayout.WEST);
f.add(b[3], BorderLayout.EAST);
f.add(b[4], BorderLayout.CENTER);
f.setSize(200,200);
f.setVisible(true);
}
public static void main(String args[]) {
new BorderExample().setup();
}
}
```

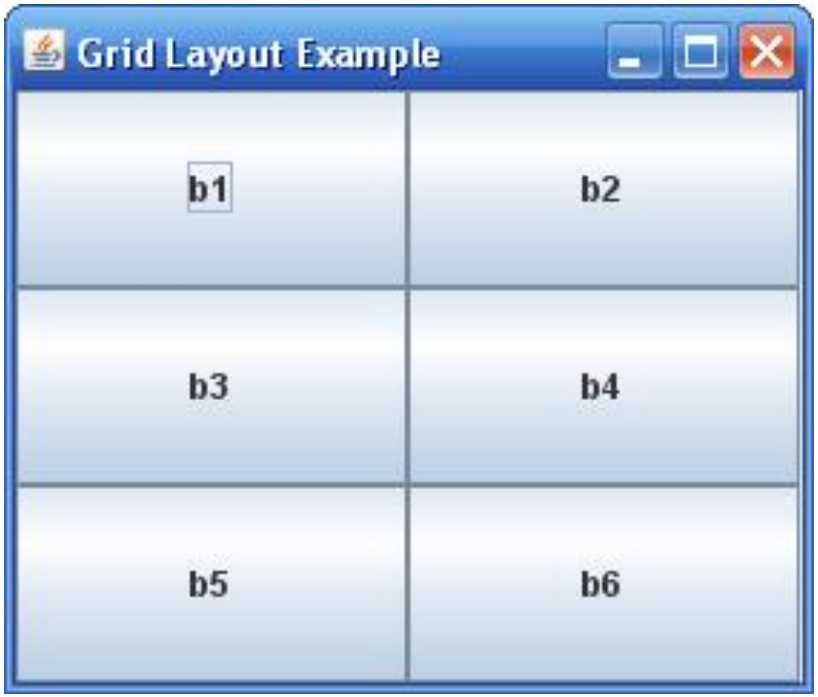


Il GridLayout

- Il GridLayout dispone i componenti da sinistra verso destra e dall'alto verso il basso all'interno di una griglia.
- Tutte le regioni della griglia hanno sempre la stessa dimensione (anche se vengono modificate le dimensioni del frame), e i componenti occuperanno tutto lo spazio possibile all'interno delle varie regioni.
- Il costruttore del GridLayout permette di specificare righe e colonne della griglia.
- Come per ilBorderLayout, i componenti occuperanno interamente le celle in cui vengono aggiunti.

Esempio

```
import java.awt.*;
import javax.swing.*;
public class GridExample {
private JFrame f;
private JButton b[]={new JButton("b1"),new
JButton("b2"),new JButton("b3"), new JButton("b4"),
new JButton("b5"), new JButton("b6")};
public GridExample() {
f = new JFrame("Grid Layout Example");
}
public void setup() {
f.setLayout(new GridLayout(3,2));
for (int i=0; i<6;++i)
f.add(b[i]);
f.setSize(200,200);
f.setVisible(true);
}
public static void main(String args[]) {
new GridExample().setup();
}
}
```



Esempio di GridLayout

GridLayoutEsempio.java

```
import javax.swing.*;
import java.awt.*;
public class GridLayoutEsempio extends JFrame {
    private static final long serialVersionUID = 1L;
    public GridLayoutEsempio() {
        initUI();
    }
    public final void initUI() {
     Container panel=this.getContentPane();
     panel.setLayout(new GridLayout(5, 4, 5, 5));
     String[] buttons = {
            "Cls", "Bck", "", "Close", "7", "8", "9", "/", "4",
            "5", "6", "*", "1", "2", "3", "-", "0", ".", "=", "+"
        };
     for (int i = 0; i < buttons.length; i++) {
            panel.add(new JButton(buttons[i]));
        }
        setTitle("GridLayout");
        setSize(350, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setVisible(true);
    }
    public static void main(String[] args) {
                new GridLayoutEsempio();
            }
}
```

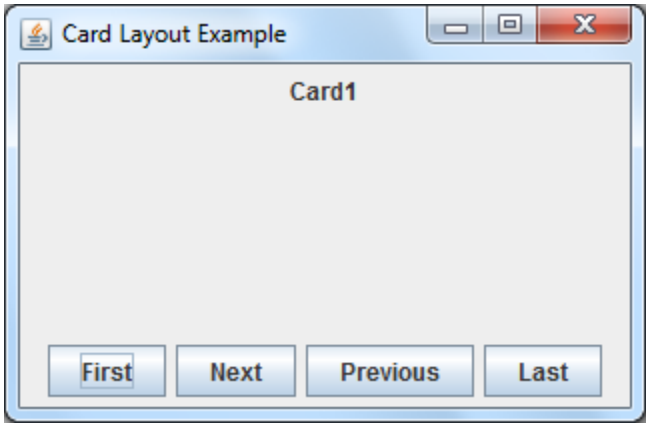
Writable | Smart Insert | 29 : 1

GridLayout

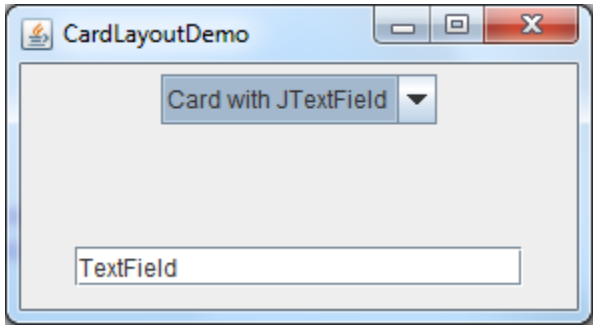
Cls	Bck		Close
7	8	9	/
4	5	6	*
1	2	3	-
0	.	=	+

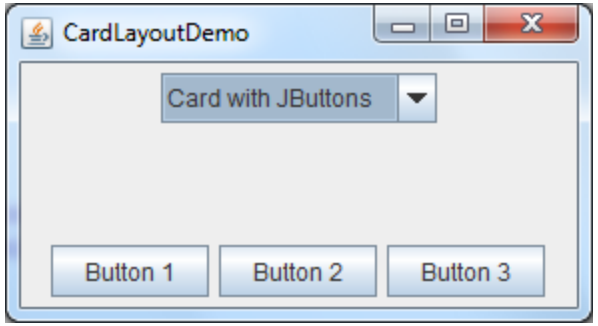
CardLayout

Permette di condividere un area mostrando un pannello alla volta di quelli aggiunti ad esso

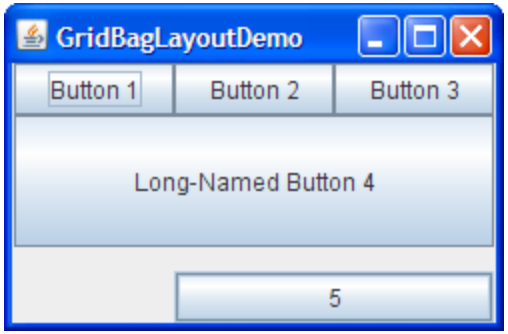


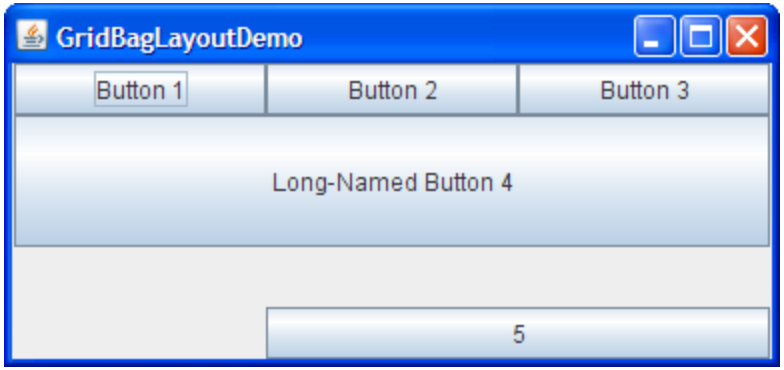
La stessa cosa è ottenibile con pannelli a schede





GridBagLayout





Creazione di una Interfaccia grafica

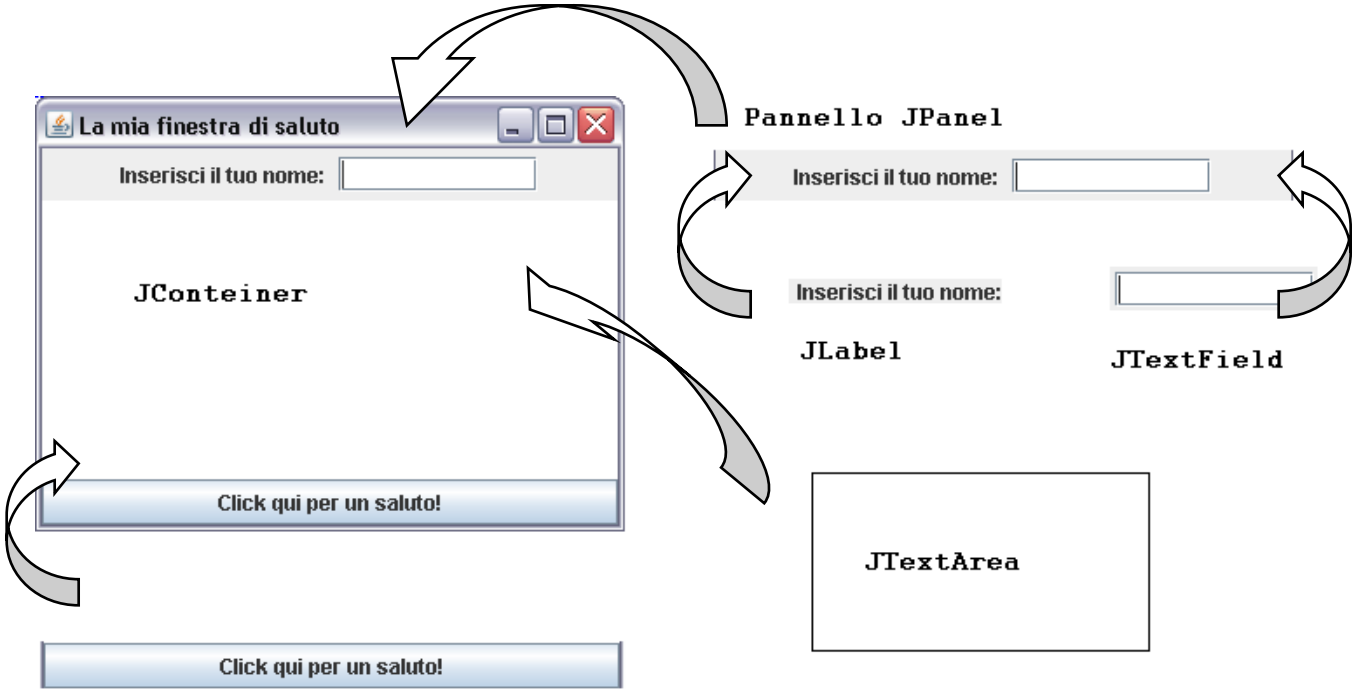
Costruiamo una prima applicazione GUI utilizzando i componenti di base per realizzare l'input e l'output e cioè una finestra nella quale l'utente potrà inserire il suo nome in una TextBox e pigiando su un bottone si visualizzera in una TextArea un saluto generato dal programma.

I componenti che utilizzeromo saranno quindi:

Una JTextField che accetterà l'input dall'utente.
Una JTextArea che visualizzerà l'output del programma.
Un JButton che permette all'utente di attivare il saluto.
Una JLabel che serve come commento di prompt per la JTextField.

Creazione di una Interfaccia grafica

La nostra finestra dovrebbe essere costruita come mostra la figura:

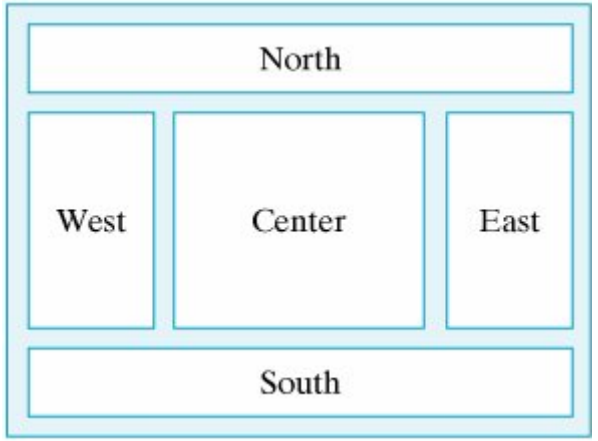


Creazione di una Interfaccia grafica

Se utilizziamo Un borderLayout

le operazioni da svolgere aranno:

- si determina un conteiner
- si costruisce un pannello
- si aggiungono componenti al pannello
- si aggiunge l'intero pannello al conteiner



Creazione di una Interfaccia grafica

```
/**
 * Inizializzazione dell'aspetto e del contenuto del frame.
 */
private void inizializza()
{

    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    pnlInput.add(prompt);
    pnlInput.add(txfNome);
    frame.getContentPane().add(BorderLayout.NORTH, pnlInput);
    frame.getContentPane().add(BorderLayout.CENTER, txaDisplay);
    frame.getContentPane().add(BorderLayout.SOUTH, btnSaluto);
    btnSaluto.addActionListener(new btnSaluto_Click());
    frame.pack();//dimensiona la finestra esattamente al suo contenuto

 }
```