

ESERCIZIO

Predisporre una maschera di input che consenta all'utente di :

- inserire articoli (e relativo prezzo), sia in lire che in euro**
- avere sempre sott'occhio il totale, sia in lire che in euro**
- poter salvare su file, in qualunque momento, l'elenco degli ordini inseriti.**

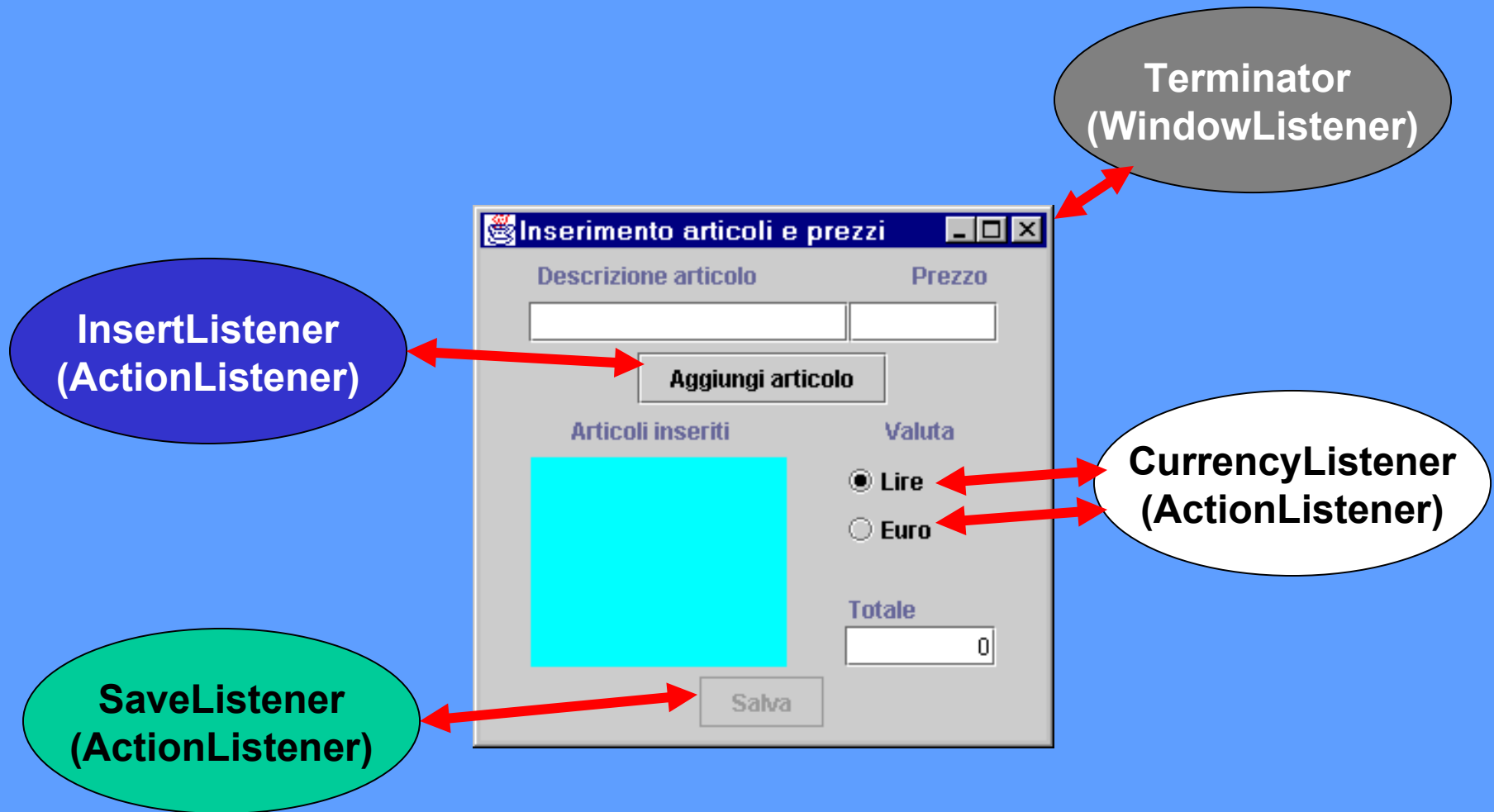
L'ASPETTO

Look & feel desiderato:

The screenshot shows a window titled "Inserimento articoli e prezzi". It has two input fields at the top: "Descrizione articolo" and "Prezzo". Below them is a button labeled "Aggiungi articolo". Under the button, there are two sections: "Articoli inseriti" which contains a large empty cyan box, and "Valuta" which has two radio buttons, "Lire" (selected) and "Euro". At the bottom right, there is a "Totale" label above a text box containing the number "0". A "Salva" button is at the bottom center.

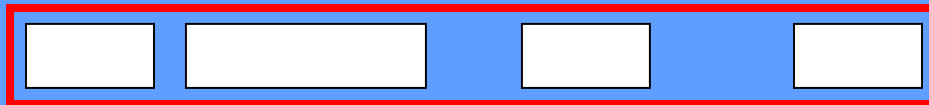
The screenshot shows the same window with data entered. The "Descrizione articolo" field is empty, and the "Prezzo" field is empty. The "Aggiungi articolo" button is present. The "Articoli inseriti" section now contains a cyan box with the text "Chitarra 1540000 ITL" and "Diapason 68 EUR". The "Valuta" section has the "Euro" radio button selected. The "Totale" text box now displays "863.34". The "Salva" button remains at the bottom.

L'ARCHITETTURA

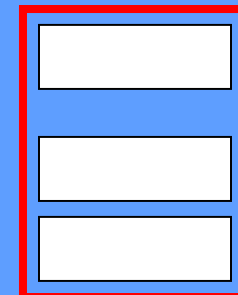


IL COMPONENTE Box

- Per disporre i componenti con ordine, è utile una **organizzazione a blocchi**: il **componente Box** serve allo scopo.
- Un **Box** è un componente *invisibile*, che organizza i suoi componenti
 - o in orizzontale (uno a fianco dell'altro)..



- ... o in verticale (uno sotto l'altro)



STRUTTURA DELL'INTERFACCIA

Struttura a blocchi:

The screenshot shows a software window titled "Inserimento articoli e prezzi". The interface is divided into several sections, with red boxes highlighting specific components:

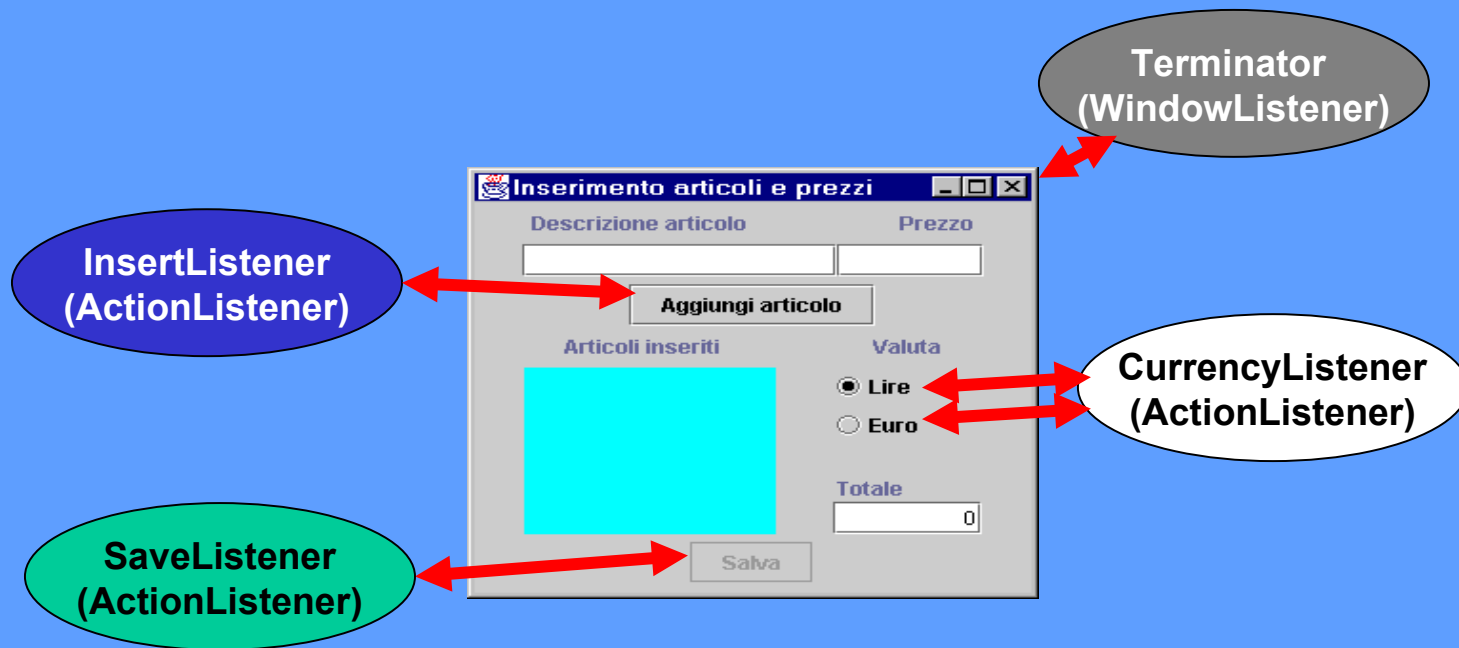
- Horizontal Boxes:** The top section contains two input fields labeled "Descrizione articolo" and "Prezzo". Below these is a button labeled "Aggiungi articolo".
- Vertical Box:** The bottom section contains a large cyan rectangular area, a radio button group for "Lire" (selected) and "Euro", and a "Totale" label with a corresponding input field showing the value "0".
- Other Elements:** A "Salva" button is located at the bottom center of the window.

Vari Box
orizzontali..

.. e un Box
verticale.

SVILUPPO DELL'APPLICAZIONE

- ① Costruzione dell'interfaccia
- ② Costruzione dei singoli listener



COSTRUZIONE DELL'INTERFACCIA

- ❶ Costruzione del primo Box orizzontale in cui collocare le due etichette
- ❷ Costruzione del secondo Box orizzontale in cui collocare i due campi di testo
- ❸ Costruzione del terzo Box orizzontale con altre etichette
- ❹ Costruzione del Box verticale per bottoni e totale
- ❺ Costruzione del quarto Box orizzontale per area di testo e Box verticale

The screenshot shows a window titled "Inserimento articoli e prezzi". The form contains the following elements:

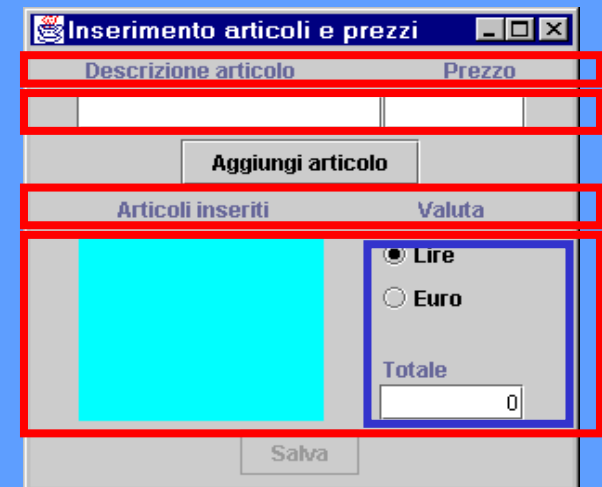
- A header row with labels "Descrizione articolo" and "Prezzo".
- Two input fields below the header.
- A button labeled "Aggiungi articolo".
- A row with labels "Articoli inseriti" and "Valuta".
- A large cyan rectangular area.
- A vertical box containing:
 - Radio buttons for "Lire" (selected) and "Euro".
 - A label "Totale" above a text input field containing the value "0".
- A "Salva" button at the bottom.

Red boxes highlight the top three horizontal sections: the header row, the input fields, and the "Articoli inseriti" / "Valuta" row. A blue box highlights the vertical box containing the currency selection and the total field.

COSTRUZIONE DELL'INTERFACCIA

Definizione dei componenti

```
public class Maschera extends JPanel {  
    JTextField nome, prezzo, totale;  
    JTextArea elenco;  
    JRadioButton lire, euro;  
    ButtonGroup grp;  
    JButton save, inserisci;  
    Box boxRadio, riga1,  
        riga2, riga3, riga4;  
    ...  
}
```



COSTRUZIONE DELL'INTERFACCIA

- ❶ Costruzione del primo Box orizzontale in cui collocare le due etichette

```
public Maschera() {  
    super();  
  
    JLabel e1 = new JLabel(  
        "Descrizione articolo");  
    JLabel etichetta2 = new JLabel("Prezzo");  
  
    riga1 = new Box(BoxLayout.X_AXIS);  
    riga1.add(etichetta1);  
    riga1.add(Box.createHorizontalStrut(80));  
    riga1.add(etichetta2);  
    ...  
}
```

uno spazio fisso e
invisibile di 80 pixel

COSTRUZIONE DELL'INTERFACCIA

- ② Costruzione del secondo Box orizzontale in cui collocare i due campi di testo

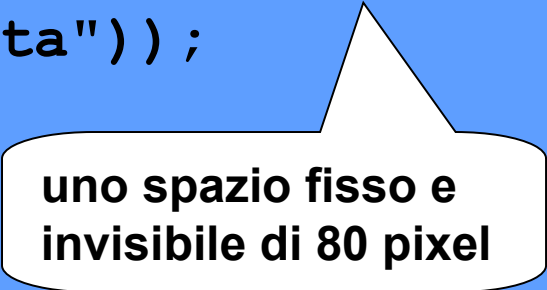
```
...  
nome = new JTextField(15);  
prezzo = new JTextField(7);  
prezzo.setHorizontalAlignment(  
                                JTextField.RIGHT);  
  
riga2 = new Box(BoxLayout.X_AXIS);  
riga2.add(nome);  riga2.add(prezzo);  
...
```

qui, nessuno spazio
intermedio esplicito.

COSTRUZIONE DELL'INTERFACCIA

③ Costruzione del terzo Box orizzontale con altre etichette

```
...  
riga3 = new Box(BoxLayout.X_AXIS);  
riga3.add(new JLabel(  
    "Articoli inseriti"));  
riga3.add(Box.createHorizontalStrut(80));  
riga3.add(new JLabel("Valuta"));  
...
```



uno spazio fisso e
invisibile di 80 pixel

COSTRUZIONE DELL'INTERFACCIA

④ Costruzione del Box verticale per bottoni e totale

```
...  
// l'area di testo  
elenco = new JTextArea(6,12);  
elenco.setEditable(false);  
elenco.setBackground(Color.cyan);  
  
// i bottoni radio  
lire = new JRadioButton("Lire", true);  
euro = new JRadioButton("Euro");  
grp = new ButtonGroup();  
grp.add(lire);   grp.add(euro);  
...
```

area di testo
non modificabile

default

COSTRUZIONE DELL'INTERFACCIA

④ Costruzione del Box verticale per bottoni e totale

```
...  
// il Box verticale vero e proprio  
boxRadio = new Box(BoxLayout.Y_AXIS);  
boxRadio.add(lire); boxRadio.add(euro);  
boxRadio.add(Box.createVerticalStrut(20));  
boxRadio.add(new JLabel("Totale"));  
totale = new JTextField("0", 5);  
totale.setHorizontalAlignment(  
    JTextField.RIGHT);  
boxRadio.add(totale);  
...
```

il totale inizialmente
riporta "0"

COSTRUZIONE DELL'INTERFACCIA

- ⑤ Costruzione del quarto Box orizzontale per area di testo e Box verticale

...

```
riga4 = new Box(BoxLayout.X_AXIS);
```

```
riga4.add(elenco);
```

```
riga4.add(Box.createHorizontalStrut(30));
```

```
riga4.add(boxRadio);
```

...

COSTRUZIONE DELL'INTERFACCIA

⑥ Assemblaggio dei vari blocchi

...

`add(riga1);`

il I° Box orizzontale

`add(riga2);`

il II° Box orizzontale

`inserisci = new JButton(
"Aggiungi articolo");`

`add(inserisci);`

il bottone "Aggiungi"

`add(riga3);`

`add(riga4);`

III° e IV° Box orizzontale

`save = new JButton("Salva");`

`save.setEnabled(false);`

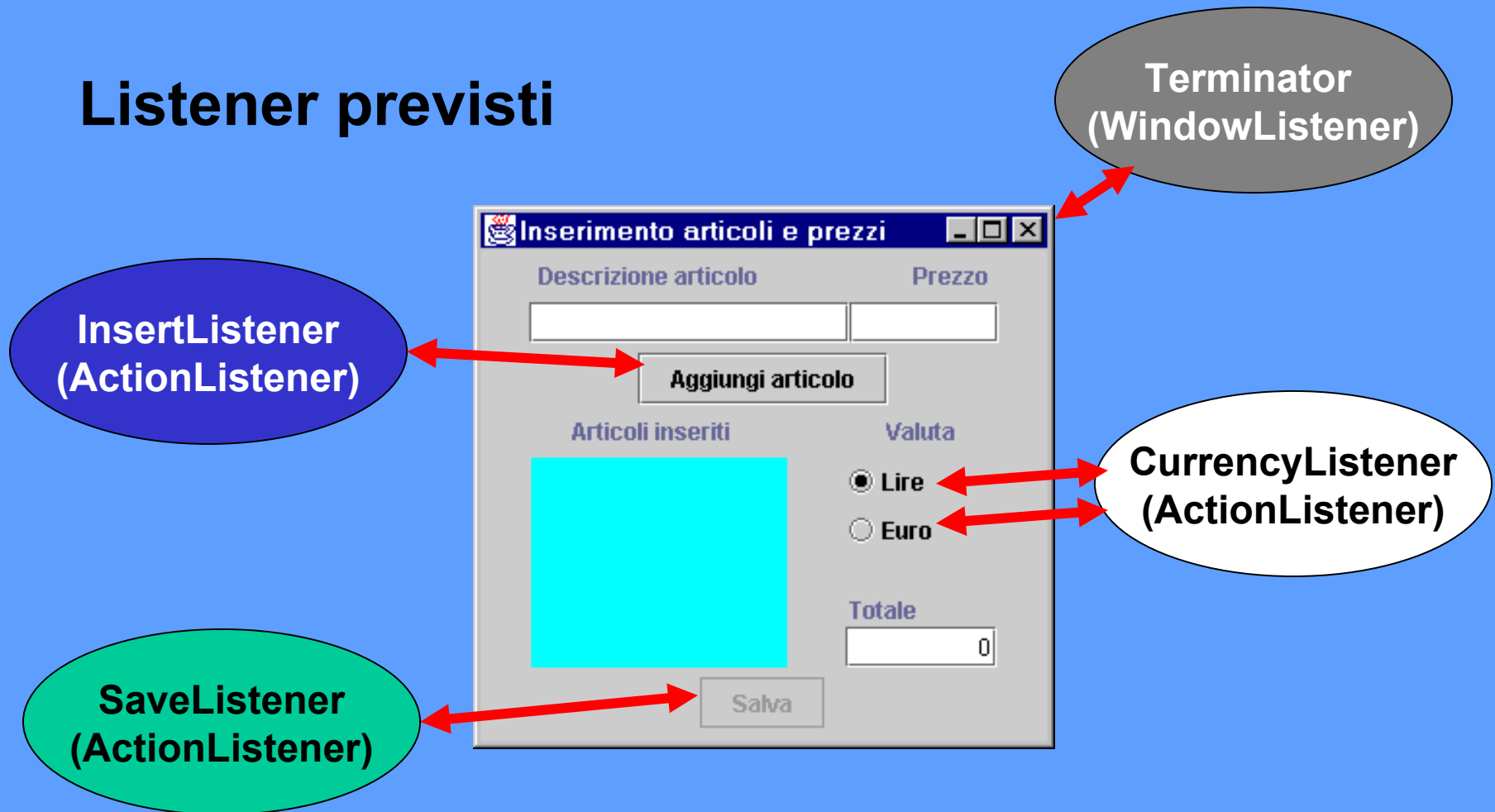
`add(save);`

il bottone "Salva"
(iniz. disabilitato)

...

COSTRUZIONE DELL'INTERFACCIA

Listener previsti



COSTRUZIONE DELL'INTERFACCIA

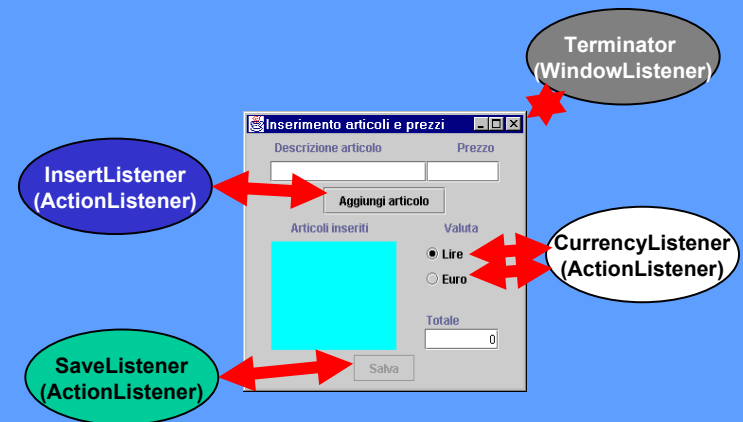
⑦ Creazione e registrazione dei listener

```
...
inserisci.addActionListener(
    new InsertListener(nome, prezzo, totale,
        elenco, lire, save));
CurrencyListener c =
    new CurrencyListener(prezzo, totale);
lire.addActionListener(c);
euro.addActionListener(c);
save.addActionListener(
    new SaveListener(elenco, totale,
        lire, save));
}
```

unico listener per
i due radiobutton

L' InsertListener

- Ascolta il pulsante "Aggiungi elemento"
- Agisce su tutti i componenti dell'interfaccia
 - recupera descrizione e prezzo
 - deduce la valuta su cui operare
 - aggiunge l'articolo all'elenco (con prezzo)
 - recupera e aggiorna il totale
 - abilita il pulsante "Salva"
- In alternativa si potevano prevedere più "mini-listener", uno per ogni funzionalità da gestire.



L' InsertListener

```
class InsertListener implements ActionListener {
    JTextField nome, prezzo, totale;
    JTextArea elenco;
    JRadioButton lire;
    JButton save;

    public InsertListener(JTextField n, JTextField p,
        JTextField t, JTextArea e, JRadioButton l,
        JButton s){
        nome=n; prezzo=p; totale=t;
        elenco=e; lire=l; save=s;
    }
    ...
}
```

L' InsertListener

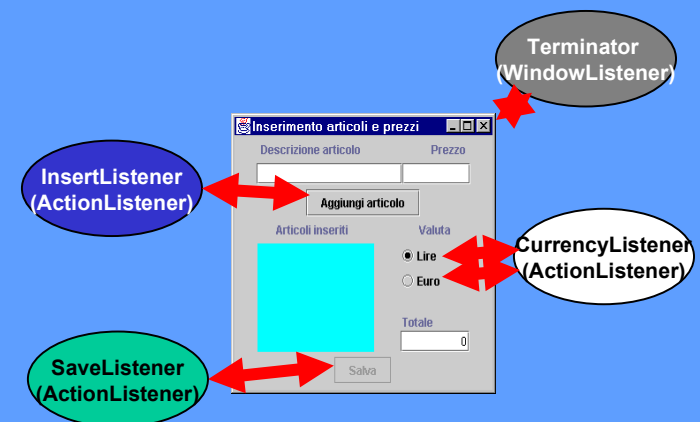
...

```
public void actionPerformed(ActionEvent e) {  
    save.setEnabled(true);  
    double tot = Double.parseDouble(totale.getText());  
    double prz = Double.parseDouble(prezzo.getText());  
    totale.setText(Double.toString(prz + tot));  
    elenco.append(nome.getText() + " " +  
        prezzo.getText() +  
        (lire.isSelected() ? " ITL" : " EUR") + "\n");  
    nome.setText("");  
    prezzo.setText("");  
    nome.requestFocus();  
}
```

**A inserimento avvenuto, è comodo
che i campi vengano puliti e il
cursore torni su "Descrizione"**

II CurrencyListener

- Ascolta i due bottoni radio (entrambi!)
- Agisce su prezzo e totale
 - li recupera, converte le stringhe in numeri...
 - ... li moltiplica o divide per il tasso di cambio...
 - ... li riconverte in stringhe e li aggiorna
- Ha il problema di capire quale dei due pulsanti è stato attivato
 - usare getSource è sicuro, ma richiede un riferimento a uno dei due pulsanti
 - usare getActionCommand è più pratico, ma impone modifiche se cambia la label.



|| CurrencyListener

```
class CurrencyListener
    implements ActionListener {
    JTextField prezzo, totale;
    public final double CAMBIO = 1936.27;

    public CurrencyListener(JTextField p,
                           JTextField t){
        prezzo=p; totale=t;
    }
    ...
}
```

II CurrencyListener

```
...  
public void actionPerformed(ActionEvent e) {  
    double tot = Double.parseDouble(totale.getText());  
    String p = prezzo.getText();  
    if (p.equals("")) p="0";  
    double price = Double.parseDouble(p);  
    if (e.getActionCommand().equals("Lire")) {  
        // erano Euro, vanno convertiti in lire  
        tot *= CAMBIO; price *= CAMBIO;  
        tot = Math rint(tot);  
        price = Math rint(price);  
    } else ...  
    ...  
}
```

Elimina la parte frazionaria di un numero reale, arrotondandolo.

II CurrencyListener

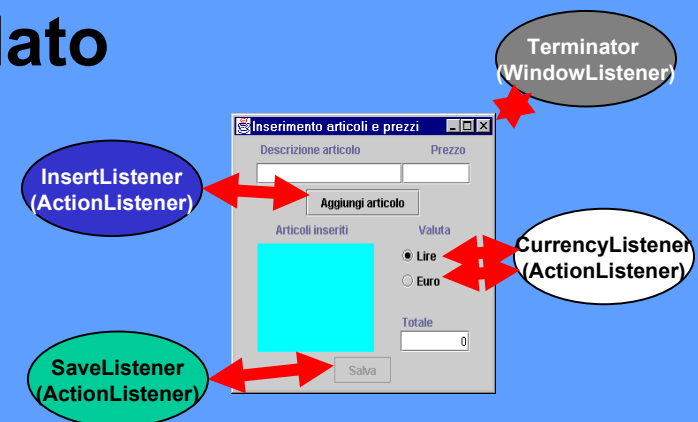
```
...  
else { // erano Lire, da convertire in Euro  
    tot /= CAMBIO; price /= CAMBIO;  
    tot = Math rint(tot*100)/100;  
    price = Math rint(price*100)/100;  
}  
totale.setText(Double.toString(tot));  
prezzo.setText(price==0.0 ? "" :  
                Double.toString(price));  
}  
}
```

Estetica: un prezzo di 0.0 non viene indicato, si mette la stringa vuota.

Arrotondamento
alla 2^a cifra decimale: si sposta la virgola, si arrotonda, e si rimette la virgola dov'era.

II SaveListener

- Ascolta il bottone "Salva"
- Agisce su elenco, totale e bottoni radio
 - recupera il testo dell'elenco e lo salva su file
 - salta una riga, recupera il totale e lo salva...
 - ... aggiungendo l'indicazione della valuta.
 - poi disabilita di nuovo il pulsante stesso
- Il nome del file si suppone dato
 - attenzione al formato dei fine linea: ' \n ' può essere tradotto in modi diversi su piattaforme diverse



|| SaveListener

```
class SaveListener implements ActionListener {  
    JTextField totale;  
    JTextArea elenco;  
    JRadioButton lire;  
    JButton save;  
  
    public SaveListener(JTextArea e, JTextField t,  
        JRadioButton l, JButton s){  
        elenco=e; totale=t; lire=l; save=s;  
    }  
    ...  
}
```

II SaveListener

```
...
public void actionPerformed(ActionEvent e) {
    double tot = Double.parseDouble(totale.getText());
    try {
        FileWriter f = new FileWriter("ELENCO.TXT");
        PrintWriter out = new PrintWriter(f);
        out.print(elenco.getText());
        out.println();
        out.print("Totale: " + totale.getText() +
            (lire.isSelected() ? " ITL" : " EUR") + "\n");
        out.println();
        out.close();
    }
    catch (IOException ex) { System.err.println(ex); }
    save.setEnabled(false);
}
}
```

Disabilita di nuovo il pulsante Salva