Le Applet Java

ed anche i gestori del layout ed altri
eventi ed altro ancora…

Le Applet

- Sono delle piccole applicazioni
- Non sono pensate per essere eseguite indipendentemente (non vengono lanciate con il main), ma da un programma esterno (browser web, appletviewer, etc.)
- **`java.applet.Applet`** è una sottoclasse di **`java.awt.Panel`**
- Restrizioni di sicurezza!!

Esempio

```
/*<APPLET CODE="CiaoATutti.class" WIDTH=150
HEIGHT=25></APPLET>*/
import java.applet.*;
import java.awt.*;

public class CiaoATutti extends Applet {
  public void paint(Graphics g) {
    g.drawString("Ciao a tutti!", 5, 25);
  }
}
```

Per scrivere un'applet…

- Occorre estendere la classe **`java.applet.Applet`**
- Occorre sovrascrivere alcuni dei suoi metodi
  - Nell'esempio abbiamo sovrascritto il metodo **`paint`** che viene invocato quando l'applet viene mostrata sullo schermo

Per visualizzare un'applet

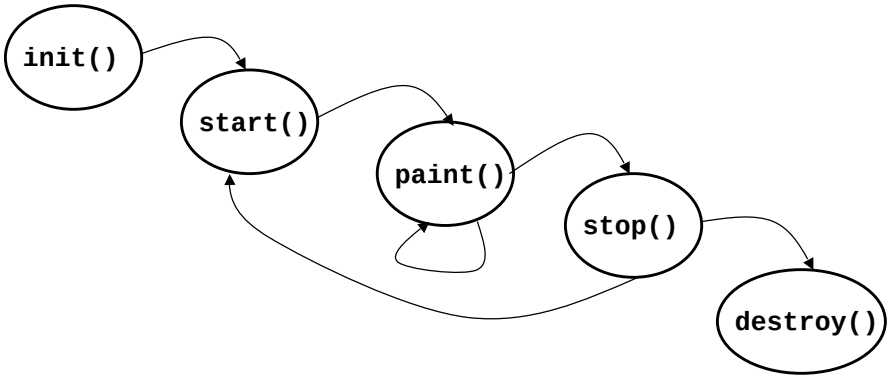
- Occorre compilare il file .java (come sempre)
- Occorre un browser web
  - In questo caso ci serve un file html
- Possiamo usare anche l'appletviewer
  - In questo caso possiamo inserire il codice html come commento all'interno del file .java (come nell'esempio)

Esempio

- Compilazione:
  - **`javac CiaoATutti.java`**
- Esecuzione
  - **`appletviewer CiaoATutti.java`**
- ma anche
  - **`appletviewer CiaoATutti.html`**
- oppure anche
  - **`appletviewer CiaoATutti.txt`**
- All'appletviewer basta che sia un file di testo con scritto da qualche parte

```
<applet code="CiaoATutti.class" width=150
height=25></applet>
```

La vita di un'Applet



Tutti i metodi sono vuoti tranne **`paint`**
che viene ereditato da **`Component`**

Parametri

• Possiamo fornire dei parametri ad un'applet

```
/*
<applet code="es06.class" width="100" height="100">
<param name="nomebottone" value="ABC">
</applet>
*/
import java.awt.*;
import java.applet.*;
public class es06 extends Applet {
  public void init() {
    String s = this.getParameter("nomebottone");
    this.add(new Button(s));
  }
}
```

Parametri 2

- I parametri si passano all'applet con il tag **`<param name="npar" value="vpar">`**
- Il metodo **`getParmeter(String npar)`** ritorna il valore dell'attributo **`value`**

Gestori del Layout

Invece di posizionare i componenti utilizzando un sistema di coordinate, in Java è possibile (e preferibile) associare al **`Container`** che contiene i vari componenti un oggetto **`LayoutManager`** che si occupa di *posizionare* e *dimensionare* i componenti secondo delle regole stabilite

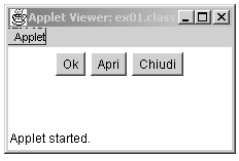
FlowLayout

- Il gestore del layout **`FlowLayout`** è quello predefinito per i **`Panel`** e le **`Applet`**
- Posiziona gli elementi come se fossero le parole di un paragrafo con allineamento centrato: un *flusso* da sinistra verso destra
- È possibile modificare l'allineamento e lo spazio orizzontale e verticale tra gli elementi

Esempio FlowLayout

```
import java.awt.*;
import java.applet.Applet;
public class ex01
              extends Applet {

  public void init() {
    add(new Button("Ok"));
    add(new Button("Apri"));
    add(new Button("Chiudi"));
  }
}
```



Come si cambia il gestore del layout

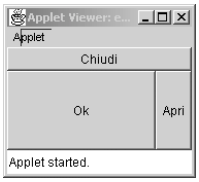
- La classe **`Container`** (e quindi tutte le sottoclassi) ha il metodo **`setLayout(LayoutManager m)`**
- Es: **`a1.setLayout(new FlowLayout())`** imposta un **`FlowLayout`** manager come gestore per il **`Container a1`**
- **`a2.setLayout(new FlowLayout(FlowLayout.LEFT))`** come prima, ma con allineamento a sinistra
- **`a3.setLayout(new FlowLayout(FlowLayout.RIGHT,20,40))`** come prima, ma con allineamento a destra, distanza orizzontale di 20 e distanza verticale di 40

BorderLayout

- È il gestore predefinito per i **`Frame`**
- Posiziona al massimo 5 elementi in 5 posizioni:
  - **`NORTH, EAST, SOUTH, WEST, CENTER`**
- I componenti vengono "stirati" per occupare tutto lo spazio a disposizione

Esempio BorderLayout

```
import java.awt.*;
import java.applet.Applet;
public class ex02
                extends Applet {
  public void init() {
    this.setLayout(new BorderLayout());
    add(new Button("Ok"),BorderLayout.CENTER);
    add(new Button("Apri"),BorderLayout.EAST);
    add(new Button("Chiudi"),BorderLayout.NORTH);
  }
}
```

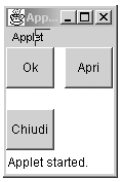


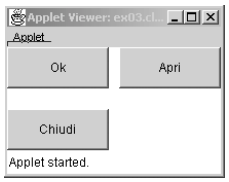
GridLayout

- Posiziona gli elementi in una griglia
- La dimensione della griglia viene definita con il costruttore **`GridLayout(int rows, int cols)`**
- I componenti vengono "stirati" per occupare tutto lo spazio a disposizione
- È possibile indicare nel costruttore anche lo spazio orizzontale e verticale (come in **`FlowLayout`**)

Esempio GridLayout

```
import java.awt.*;
import java.applet.Applet;
public class ex03
              extends Applet {
  public void init() {
    this.setLayout(new GridLayout(2,2,10,20));
    add(new Button("Ok"));
    add(new Button("Apri"));
    add(new Button("Chiudi"));
  }
}
```





Movimento del Mouse

- Un ascoltatore degli eventi relativi al movimento del mouse deve implementare l'interfaccia **`MouseMotionListener`**
- Ha due metodi:
  - **`public void mouseDragged(MouseEvent e)`**
  - **`public void mouseMoved(MouseEvent e)`**
- Esiste anche un adattatore **`MouseMotionAdapter`**

Eventi della tastiera

- L'interfaccia **`KeyListener`** definisce i metodi per gestire gli eventi della tastiera
  - **`public void keyTyped(KeyEvent e)`**
  - **`public void keyPressed(KeyEvent e)`**
  - **`public void keyReleased(KeyEvent e)`**
- Esiste anche un adattatore **`KeyAdapter`**

Focus

- Un componente ha il focus quando è pronto per ricevere l'input (ad esempio quando si clicca su una casella di testo)
- L'interfaccia **`FocusListener`** definisce i metodi per gestire l'acquisto e la perdita del focus
  - **`void focusGained(FocusEvent e)`**
  - **`void focusLost(FocusEvent e)`**
- Esiste l'adattatore **`FocusAdapter`**

Euro convertitore

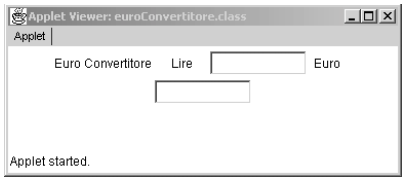
- Obiettivo: scrivere un'applet che permetta di fare le conversioni lire – euro e viceversa

Euro convertitore 2 (componenti)

- Per prima cosa pensiamo alla parte grafica, poi passeremo alla gestione degli eventi
- Abbiamo bisogno di
  - Un'etichetta (con la scritta "Euro Convertitore")
  - Due caselle di testo e due etichette, una con la scritta "lire" e l'altra con la scritta "euro"

Euro convertitore 3

```
/*<applet code="euroConvertitore.class" width="380"
height="100">
</applet>*/
import java.awt.*;
import java.applet.*;
public class euroConvertitore extends Applet {
  public void init() {
     this.add(new Label("Euro Convertitore"));
     TextField lire = new TextField(10);
     TextField euro = new TextField(10);
     this.add(new Label("Lire"));
     this.add(lire);
     this.add(new Label("Euro"));
     this.add(euro);
  }
}
```



Euro convertitore 4 (layout)

- Per aggiustare il layout impostiamo il gestore a BorderLayout
- Aggiungiamo a north un Panel (che verrà "stirato") e a questo panel aggiungiamo la Label "Euro Convertitore
- Aggiungiamo un altro Panel al centro e poi a questo le restanti due etichette e caselle di testo

Euro convertitore 5

```
…
  public void init() {
     this.setLayout(new BorderLayout());
     Panel p = new Panel();
     Panel p2 = new Panel();
     p2.add(new Label("Euro Convertitore"));
     this.add(p2, BorderLayout.NORTH);
     this.add(p, BorderLayout.CENTER);
     TextField lire = new TextField(10);
     TextField euro = new TextField(10);
     p.add(new Label("Lire"));
     p.add(lire);
     p.add(new Label("Euro"));
     p.add(euro);
  }
}
```

Euro convertitore 6 (eventi)

- Vogliamo che quando si scrive nella casella di testo delle lire automaticamente venga aggiornata la casella degli euro
- L'evento da gestire è il cambiamento del testo e quindi dobbiamo scrivere un TextListener

Euro convertitore 7

```
class convertitore implements TextListener {
  double tasso;
  TextComponent out;
  convertitore(TextComponent output, double tasso) {
     this.out = output;       this.tasso = tasso;
  }
  public void textValueChanged(TextEvent e) {
     TextComponent c = (TextComponent)e.getSource();
     double d = 0;
     String s = "";
     d = Double.parseDouble(c.getText());
     s += (d*tasso);
     out.setText(s);
  }
}
```

Euro convertitore 8

```
public class euroConvertitore extends Applet {
  public void init() {
  …
     TextField lire = new TextField(10);
     TextField euro = new TextField(10);
     convertitore l2e = new convertitore(euro,(1/1936.27));
     convertitore e2l = new convertitore(lire,1936.27);
     lire.addTextListener(l2e);
     euro.addTextListener(e2l);
  …
  }
}
```

Euro convertitore 9 (problemi)

- Non abbiamo tenuto conto di
  - Errori di calcolo ed arrotondamenti
  - Errori inseriti dall'utente
  - Ogni volta che si cambia il testo in una casella viene lanciato un TextEvent che viene catturato da un TextListener il quale a sua volta cambia il testo nell'altra casella che, a sua volta, lancerà un altro TextEvent e così via in un loop infinito

Euro convertitore 10 (loop)

- Per risolvere il problema del loop di eventi facciamo in modo che la classe ascoltatore cambi il testo della casella di output, solo se oltre ad avere il testo modificato ha anche il focus

Euro convertitore 11

```
class convertitore implements TextListener, FocusListener {
  double tasso;
  TextComponent out;
  boolean editing = false;
  …
  public void textValueChanged(TextEvent e) {
     …
     if (editing)
        out.setText(s);
  }
  public void focusGained(FocusEvent e) {
     editing = true;
  }
  public void focusLost(FocusEvent e) {
     editing = false;
  }
}
```

Euro convertitore 12

```
public class euroConvertitore extends Applet {
  public void init() {
  …
     lire.addTextListener(l2e);
     euro.addTextListener(e2l);
     lire.addFocusListener(l2e);
     euro.addFocusistener(e2l);
  …
  }
}
```

Euro convertitore 13 (errori)

- Se l'utente sbaglia ad inserire i dati il metodo Double.parseDouble lancia un'eccezione NumberFormatException. Catturiamo l'eccezione e scriviamo "errore" nella casella di testo
- Infine scriviamo una funzione per arrotondare ad un certo numero di cifre decimali

Euro convertitore 14

```
class convertitore implements TextListener, FocusListener {
  …
  public void textValueChanged(TextEvent e) {
     …
     double d = 0;  String s = "";
     try {
        d = Double.parseDouble(c.getText());
        s += (this.arrotonda(d*tasso,16));
     } catch(NumberFormatException ex) {s = "errore!";}
     …
  }
  …
  double arrotonda(double d, int p) {
     return Math.rint(d*Math.pow(10,p))/Math.pow(10,p);
  }
}
```