Fondamenti di informatica
un approccio a oggetti con Java

Luca Cabibbo

Applet

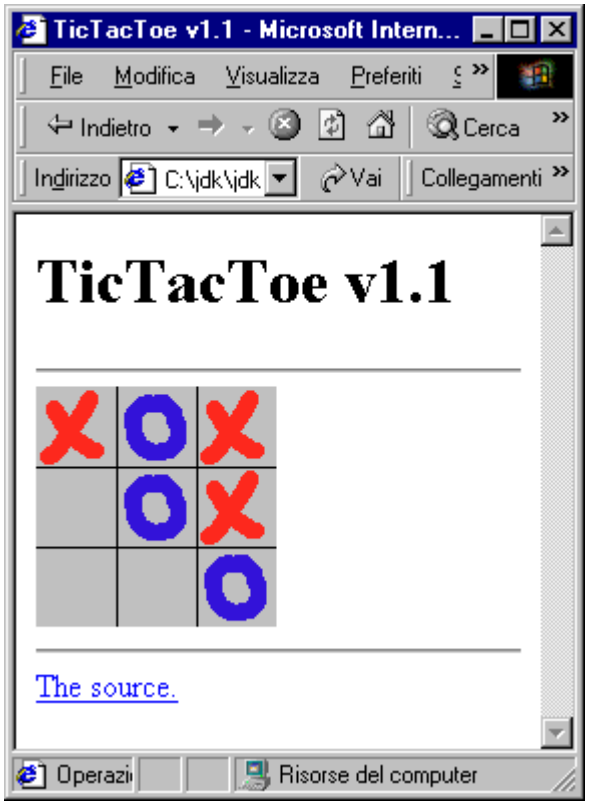
Capitolo 26

settembre 2001

Contenuti

- Introduzione agli applet
  - la classe **Applet**
  - elementi di grafica
  - la classe **JApplet**
- Ciclo di vita di un applet
  - pietre miliari nella vita di un applet
  - metodi di un applet

Introduzione



Gli applet sono una tipologia di programmi Java che, a differenza delle applicazioni Java, sono realizzati non per essere eseguiti autonomamente, ma piuttosto per essere immersi in altre applicazioni

- in particolare, gli applet possono essere immersi in pagine web ed eseguiti da browser web

Introduzione agli applet

Un **applet** è un (piccolo) programma realizzato per essere immerso (ovvero, eseguito) nell'ambito di un'altra applicazione

- gli applet sono solitamente immersi nelle pagine web (nei documenti HTML)
  - i principali browser web sono dotati di una macchina virtuale Java e sono in grado di eseguire applet
  - se una pagina web contiene un applet, un'area della pagina è dedicata alla visualizzazione dell'applet
- in contrapposizione, le applicazioni Java sono programmi che vengono eseguiti autonomamente

Per realizzare applet è utile comprenderne le caratteristiche principali, ed in particolare l'interazione tra applet e i browser web che li eseguono

Un primo esempio — l'applet scrittore

```
import java.applet.Applet;
import java.awt.Graphics;

/** Applet che visualizza una frase. */
public class AppletScrittore extends Applet {
    /* Visualizza una frase. */
    public void paint(Graphics g) {
        g.drawString("Il mio primo applet", 25, 50);
    }
}
```

Questa è la definizione di un semplice applet che visualizza una frase (nell'area della pagina web dedicata alla sua visualizzazione)

- per realizzare un applet è necessario definire una classe che estende la classe **Applet** (del package **java.applet**)
- la classe che implementa un applet non contiene il metodo di classe **void main(String[] args)** (che va definito per le classi applicazioni) ma piuttosto un certo numero di metodi d'istanza

La classe Applet

In generale, un applet è un oggetto istanza che, polimorficamente, è di tipo **Applet**

- la classe **Applet** del package **java.applet** è il progetto di un applet "primordiale", elementare, che apparentemente non fa nulla
- per realizzare un nuovo applet è necessario estendere la classe **Applet** per aggiungere comportamento all'applet primordiale

In pratica, nella classe **Applet** sono definiti i metodi con cui i browser web (o meglio, le macchine virtuali Java di cui i browser web sono dotati) possono interagire con un applet, e che vanno eventualmente ridefiniti per descrivere che cosa deve fare l'applet in corrispondenza alle richieste del browser web che lo esegue

- ad esempio, quando un browser web deve eseguire un applet, per prima cosa istanzia l'applet e, prima o poi, gli chiede di eseguire il metodo **paint(...)**

Interazione tra un browser e un applet

Per interagire con una classe applicazione, è possibile usare il comando **java** che

- avvia una macchina virtale Java
- costruisce un oggetto classe dalla classe specificata
- richiede a questo oggetto classe di eseguire il metodo **main(…)**

Come si interagisce con un applet?

- una pagina web può contenere un applet, se nella pagina è specificato, mediante un elemento **APPLET**, il nome della classe che definisce l'applet
- in questo caso, il browser web
  - scarica il bytecode della classe che è stata specificata
  - istanzia un nuovo oggetto da questa classe
  - richiede a questo oggetto di eseguire alcuni metodi, in un modo che dipende da come l'utente del browser interagisce con il browser

Applet e pagine Web

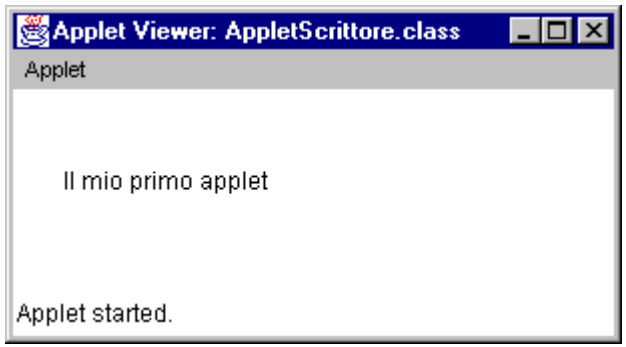
Per indicare la presenza di un applet in una pagina web va usato l'elemento **APPLET**, come nel seguente esempio

```
<HTML>
  <HEAD><TITLE>L'applet scrittore</TITLE></HEAD>
  <BODY>
    <APPLET CODE="AppletScrittore.class"
            WIDTH=300 HEIGHT=100>
    </APPLET>
  </BODY>
</HTML>
```

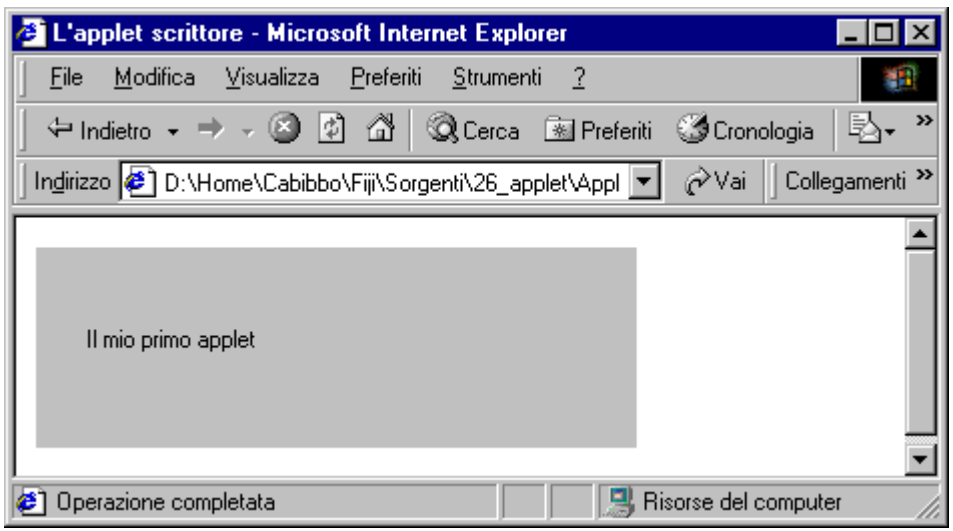
- un ruolo fondamentale della pagina web che immerge un applet è quello di assegnare all'applet un'area rettangolare dedicata alla visualizzazione dell'applet
- in particolare, un elemento **APPLET** deve contenere almeno
  - un attributo **CODE** che specifica il nome dell'applet
  - gli attributi **WIDTH** e **HEIGHT** che specificano, rispettivamente, la larghezza e l'altezza dell'area dedicata all'applet

Esecuzione dell'applet scrittore

Esecuzione mediante **appletviewer** (parte del Java 2 SDK)



Esecuzione mediante un browser web



Il metodo paint(...)

Come è stato già accennato, il browser web, quando esegue un applet, prima o poi gli chiede di eseguire il metodo **paint(...)**

- il metodo **paint(...)** viene invocato ed eseguito ogni volta che è necessario visualizzare o aggiornare l'area dedicata alla visualizzazione dell'applet
  - questo avviene, ad esempio, nel momento in cui il browser web deve visualizzare l'applet
- per default, il metodo **paint(...)** (di **Applet**) non fa nulla
  - tuttavia, è possibile ridefinire il comportamento del metodo **paint(...)** per far visualizzare qualcosa all'applet

Il metodo void paint(Graphics g)

Gli oggetti istanza della classe **Graphics** (del package **java.awt**) modellano oggetti grafici, ovvero oggetti che sono in grado di eseguire operazioni di disegno

- normalmente, un oggetto **Graphics** è associato a un oggetto con capacità grafiche (come ad esempio a un applet) ed è chiamato un **contesto grafico** dell'oggetto a cui è associato

Quando il browser richiede a un applet di eseguire il metodo **void paint(Graphics g)**

- gli passa come parametro il riferimento al contesto grafico **g** associato all'applet
- il metodo **paint** contiene normalmente istruzioni che richiedono a **g** di disegnare o di visualizzare stringhe, al fine di visualizzare lo stato corrente dell'applet

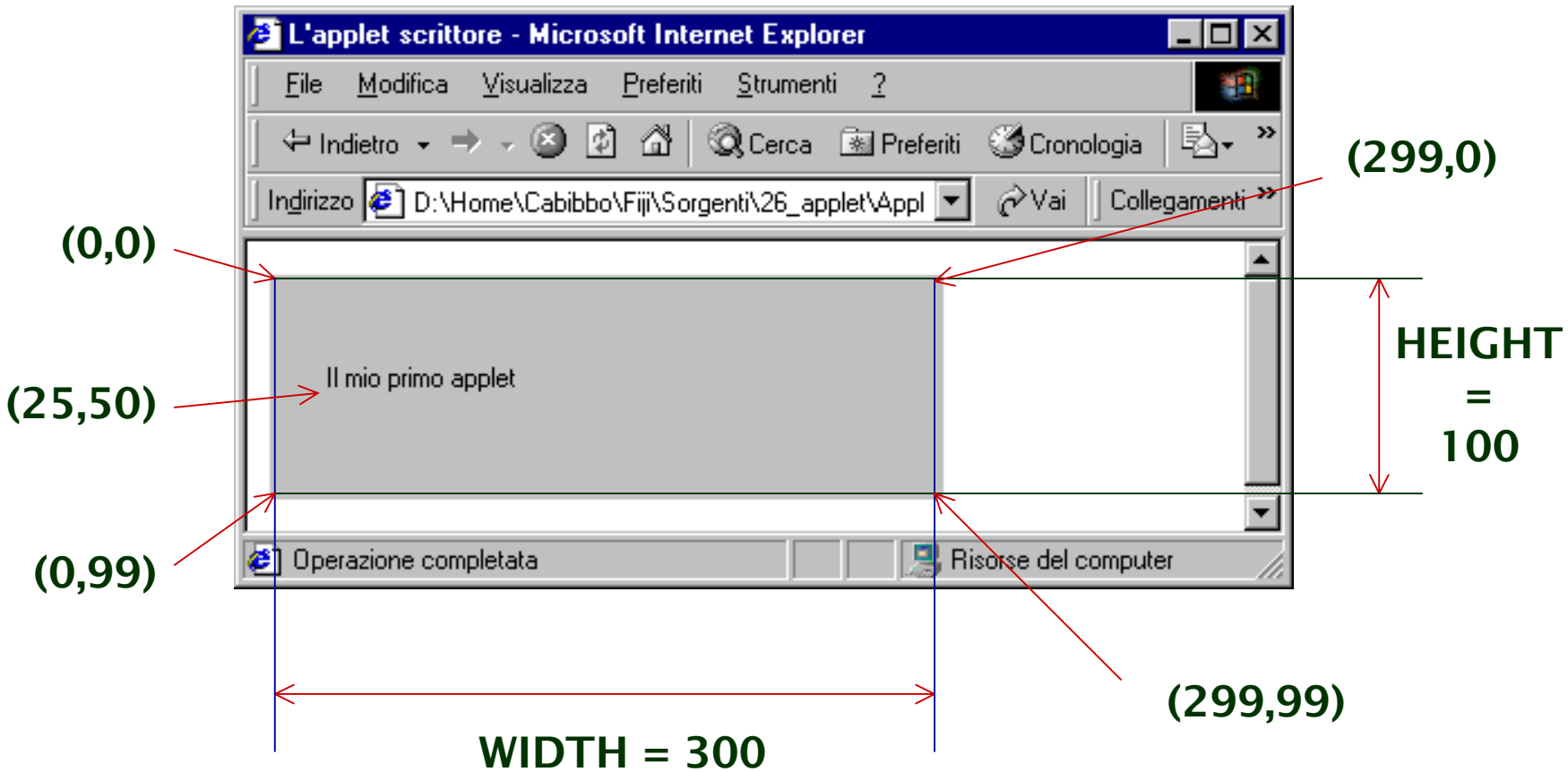
Elementi di grafica

A un contesto grafico normalmente è associata un'area rettangolare in un piano

- i punti di tale rettangolo sono identificati da una coppia di coordinate intere, espresse in "punti" (i punti sono quindi l'unità di misura usata per la grafica)
- le coordinate sono relative all'origine del contesto grafico, localizzata nel suo angolo in alto a sinistra
- per un punto di coordinate **(X,Y)**
  - la prima coordinata **X** indica uno spostamento orizzontale verso destra
  - la seconda coordinata **Y** indica uno spostamento verticale verso il basso
- in un rettangolo di larghezza **L** e altezza **H**
  - l'angolo in lato a sinistra ha coordinate **(0,0)**
  - l'angolo in basso a destra ha coordinate **(L-1,H-1)**

Elementi di grafica

La seguente figura esemplifica il sistema di coordinate in un applet



Principali metodi della classe Graphics

Ecco alcuni metodi della classe **Graphics**

- **void drawLine (int x1, int y1, int x2, int y2)**
  - disegna una linea tra i punti (**x1**, **y1**) e (**x2**, **y2**)
- **void drawRect (int x, int y, int w, int h)**
  - disegna il contorno di un rettangolo con estremo in alto a sinistra nel punto **(x, y)**, larghezza **w+1** e altezza **h+1**
  - l'estremo in basso a destra del rettangolo avrà coordinate **(x+w, y+h)**
- **void fillRect (int x, int y, int w, int h)**
  - disegna un rettangolo (pieno) con estremo in alto a sinistra nel punto **(x, y)**, larghezza **w** e altezza **h**
  - l'estremo in basso a destra del rettangolo avrà coordinate **(x+w-1, y+h-1)**
- **void drawString (String s, int x, int y)**
  - visualizza la stringa **s** — la base (in basso a sinistra) della stringa è posta nel punto di coordinate **(x, y)**

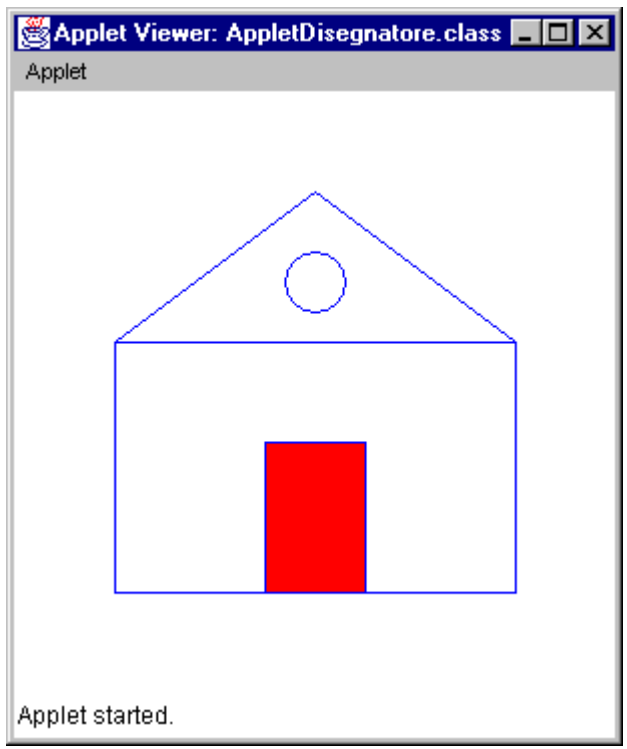
Principali metodi della classe Graphics

- **void drawOval (int x, int y, int w, int h)**
  - disegna il contorno di un ovale contenuto nel rettangolo con estremo in alto a sinistra nel punto **(x, y)**, larghezza **w+1** e altezza **h+1**
- **void fillOval (int x, int y, int w, int h)**
  - disegna un ovale (pieno) contenuto nel rettangolo con estremo in alto a sinistra nel punto **(x, y)**, larghezza **w** e altezza **h**
- **void drawPolyline (int[] xP, int[] yP, int nP)**
  - disegna una linea spezzata (aperta) composta da **nP** punti di coordinate rispettivamente **(xP[0], yP[0])**, **(xP[1], yP[1])**, **…, (xP[nP–1], yP[nP–1])**
- **void drawPolyline (int[] xP, int[] yP, int nP)**
  - disegna una linea spezzata (chiusa) composta da **nP** punti di coordinate **(xP[0], yP[0])**, **…,(xP[nP–1], yP[nP–1])**
- **void setColor (Color c)**
  - modifica il colore delle successive attività di disegno

Un applet disegnatore

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;

/* Applet che disegna una casetta (in un'area 300x300 punti). */
public class AppletDisegnatore extends Applet {
    /* Visualizza una casetta. */
    public void paint(Graphics g) {
        /* il colore delle linee è blu */
        g.setColor(Color.blue);
        /* disegna il corpo della casetta */
        g.drawRect(50, 125, 200, 125);
        /* disegna il tetto */
        g.drawLine(50, 125, 150, 50);
        g.drawLine(250, 125, 150, 50);
        /* disegna un abbaino */
        g.drawOval(135, 80, 30, 30);
        /* disegna la porta */
        g.drawRect(125, 175, 50, 75);
        g.setColor(Color.red);
        g.fillRect(126, 176, 49, 74);
    }
}
```



La classe JApplet

Il linguaggio Java permette la definizione di applet fin dalla sua prima versione, mediante una tecnologia chiamata AWT (Abstract Window Toolkit)

- tuttavia, in successive versioni di Java (in particolare, della sua libreria di classi) è stata introdotta una nuova tecnologia, chiamata JFC/Swing, di supporto alla realizzazione di interfacce grafiche e di applet
- in particolare, la classe **JApplet** (del package **javax.swing**) è una versione estesa della classe **Applet**, in grado di supportare componenti Swing
  - la classe **JApplet** estende la classe **Applet**

Data la rilevanza dell'architettura Swing, d'ora in poi sarà fatto riferimento a questa tecnologia anziché alla tecnologia AWT

Da Applet a JApplet

Il seguente esempio illustra la definizione di un **JApplet**

```
import javax.swing.JApplet;
import java.awt.Graphics;

/** Applet (Swing) che visualizza una frase. */
public class AppletScrittoreSwing extends JApplet {
    /* Visualizza una frase. */
    public void paint(Graphics g) {
        g.drawString("Il mio primo applet Swing", 25, 50);
    }
}
```

Principali differenze tra applet AWT e applet Swing

- per definire un applet Swing
  - va importata la classe **javax.swing.JApplet** anziché la classe **java.awt.Applet**
  - va estesa la classe **JApplet** anziché la classe **Applet**
- non tutti i browser web supportano gli applet Swing

Ciclo di vita di un applet

Un applet è un oggetto, che vive in (ovvero, viene eseguito da) una macchina virtuale Java di un browser web

- vengono ora descritte le “pietre miliari” (milestones) nella vita di un applet
- ciascuna pietra miliare nella vita di un applet corrisponde a un evento che si verifica nel browser e che comporta, contestualmente, la richiesta all’applet di eseguire una operazione (invocazione di metodo)
- scrivere un applet vuol dire principalmente definire i metodi la cui esecuzione viene richiesta in corrispondenza a questi eventi fondamentali

Pietre miliari nella vita di un applet

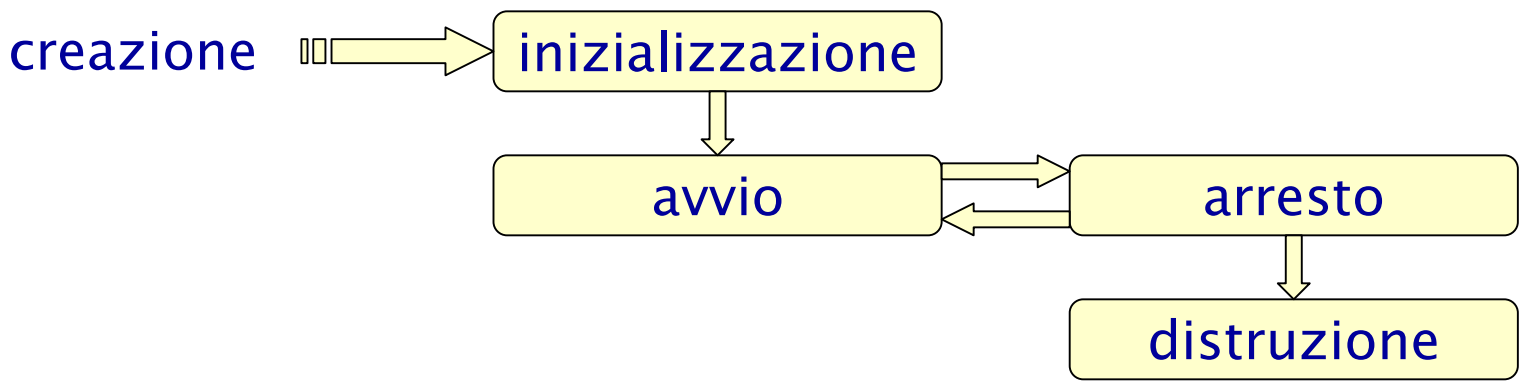
Le quattro pietre miliari principali nella vita di un applet (nell'ambito della pagina in cui è immerso) sono

- **inizializzazione**
  - quando il browser deve visualizzare un applet, ne scarica il bytecode della classe (che deve estendere la classe **Applet**), crea un oggetto da questa classe e gli chiede di inizializzare il proprio stato
  - l'inizializzazione avviene una sola volta nella vita di un applet
- **avvio**
  - subito dopo la creazione e l'inizializzazione, viene avviata l'esecuzione dell'applet
  - se la pagina contenente l'applet viene lasciata e poi viene nuovamente visitata (perché viene visitata un'altra pagina, oppure perché la pagina viene iconizzata e poi de-iconizzata), l'esecuzione dell'applet viene nuovamente ri-avviata (ma l'applet non viene ri-inizializzato)

Pietre miliari nella vita di un applet

- **arresto**
  - l'esecuzione di un applet viene arrestata quando la pagina viene lasciata (perché viene visitata un'altra pagina, oppure il browser viene iconizzato oppure chiuso)
- **distruzione**
  - un applet viene distrutto quando il browser viene chiuso (la distruzione di un applet segue l'arresto della sua esecuzione)

La seguente figura mostra gli stati in cui può trovarsi un applet e le possibili transizioni tra i vari stati



Metodi di un applet

In corrispondenza alle quattro pietre miliari della vita di un applet (inizializzazione, avvio, arresto e distruzione) all'applet viene richiesta rispettivamente l'esecuzione dei metodi **void init()**, **void start()**, **void stop()** e **void destroy()**

- nella classe **Applet**, questi metodi non fanno nulla
- è tuttavia possibile ri-definire uno o più di questi metodi per caratterizzare il comportamento dell'applet che si vuole realizzare
  - infatti, gli applet vengono definiti estendendo la classe **Applet** o **JApplet**

Vita di un applet

Viene ora riassunta la vita di un applet

Caricamento dell'applet

- avviene quando il browser deve visualizzare un applet
  - viene creata una istanza **a** dell'applet
  - l'applet viene inizializzato — **a.init()**
  - viene avviata l'esecuzione dell'applet — **a.start()**

Sospensione dell'applet

- avviene quando il browser deve accedere a un'altra pagina, o quando viene iconizzato
  - l'esecuzione dell'applet viene arrestata — **a.stop()**

Ri-attivazione dell'applet

- avviene quando il browser accede nuovamente a una pagina contenente un applet già inizializzato
  - l'esecuzione dell'applet viene ri-avviata — **a.start()**

Vita di un applet

Distruzione dell'applet

- avviene quando il browser viene chiuso, o reclama le risorse impegnate dall'applet
  - l'esecuzione dell'applet viene arrestata — **a.stop()** — se questo non era ancora avvenuto
  - l'applet viene distrutto — **a.destroy()**

Il metodo void paint(Graphics g)

Esistono altre pietre miliari nella vita degli applet, di importanza minore rispetto alle quattro fondamentali, ma utili in pratica

- in particolare, una pietra miliare è quella del **disegno**
  - ogni volta che il browser ha bisogno di visualizzare o ri-visualizzare l'applet (ad esempio, subito dopo la chiusura o lo spostamento di una finestra che copriva il browser), all'applet viene chiesto di disegnarsi
  - **void paint(Graphics g)** è il metodo che implementa l'operazione di disegno di un applet
- l'operazione di disegno segue sempre quella di avvio
  - inoltre è possibile che un applet venga disegnato più volte nel corso della sua vita

Il metodo void init()

Il metodo **void init()** viene usato per inizializzare l'applet

- normalmente, il metodo **init()** si deve curare dell'inizializzazione dello stato (delle variabili d'istanza) dell'oggetto applet, anche mediante la creazione di altri oggetti eventualmente utilizzati dall'applet (ad esempio, di altri componenti grafici)

Per gli applet non sono solitamente previsti costruttori

- l'inizializzazione che viene solitamente svolta da un costruttore, avviene per gli applet nell'ambito del metodo **init()**
- il metodo **init()** è responsabile dell'inizializzazione dell'applet
- tuttavia, il metodo **init()** non è responsabile dell'avvio delle azioni che devono essere svolte dall'applet

Il metodo void start()

Il metodo **void start()** descrive le azioni che devono essere svolte dall'applet

- il metodo **start()** si cura della gestione degli oggetti creati dal metodo **init()**
- spesso, il metodo **start()** deve semplicemente avviare dei nuovi thread di esecuzione
- d'altra parte, il metodo **start()** non si deve curare della visualizzazione degli oggetti che gestisce — di questo si occupa il metodo **paint(...)**

I metodi void stop() e void destroy()

Il metodo **void stop()** descrive le azioni che devono essere svolte quando viene sospesa l'esecuzione dell'applet

- il metodo **stop()** viene solitamente ridefinito solo se è stato ridefinito anche il metodo **start()**
- il metodo **stop()** deve interrompere le azioni avviate dal metodo **start()**
- spesso, il metodo **stop()** deve semplicemente arrestare i thread di esecuzione avviati dal metodo **start()**

Il metodo **void destroy()** descrive le azioni che devono essere svolte quando l'applet viene definitivamente interrotto

- il metodo **destroy()** viene solitamente usato per distruggere i thread di esecuzione avviati dal metodo **start()**

Esempio — un applet con parametri

Il seguente esempio mostra l'uso del metodo **init()** e delle variabili d'istanza di un applet

- l'esempio è basato sulle seguenti considerazioni
  - la pagina web che contiene un applet può specificare parametri per l'applet, mediante elementi **PARAM**
  - un applet può accedere ai parametri specificati dalla pagina web che lo contiene
  - un applet può accedere alle dimensioni dell'area dedicata alla sua visualizzazione dalla pagina web che lo contiene
- l'applet **AppletConParametri** visualizza le sue dimensioni, insieme ad una stringa specificata come parametro dalla pagina web che lo contiene
- viene utilizzata una pagina web contenente due diversi applet **AppletConParametri**

Applet con parametri

```
<HTML>
  <HEAD>
    <TITLE>Applet con parametri</TITLE>
  </HEAD>
  <BODY>
    <P>Vengono create due istanze dell'applet
      AppletConParametri.</P>
    <APPLET CODE="AppletConParametri.class"
            WIDTH=200 HEIGHT=100>
      <PARAM NAME=descrizione VALUE="Applet largo">
    </APPLET>
    <APPLET CODE="AppletConParametri.class"
            WIDTH=100 HEIGHT=200>
      <PARAM NAME=descrizione VALUE="Applet alto">
    </APPLET>
  </BODY>
</HTML>
```

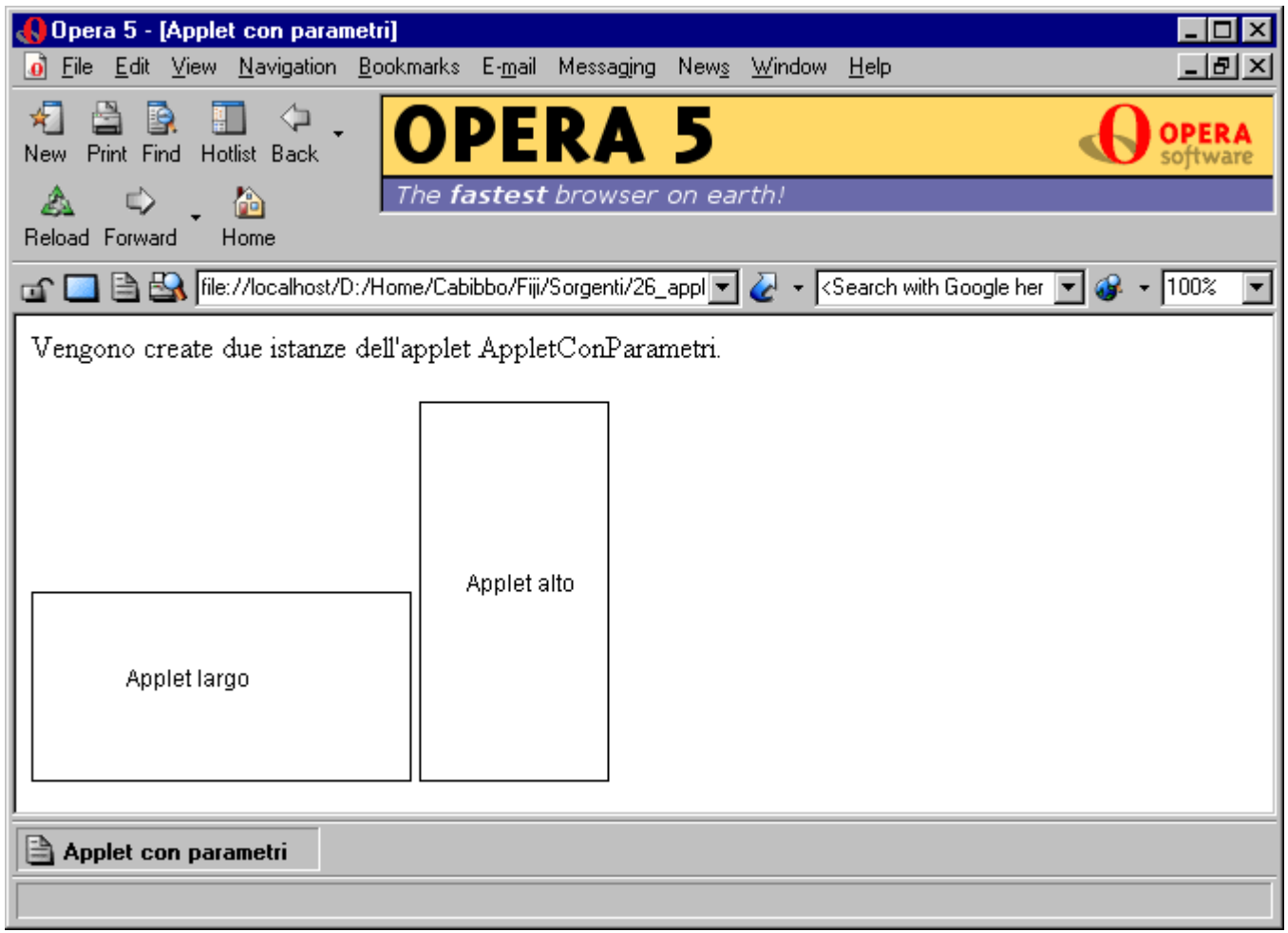
Applet con parametri

```
import javax.swing.JApplet;
import java.awt.Graphics;

/* Applet che visualizza la sua descrizione (specificata
 * come parametro della pagina web che lo contiene) in
 * un rettangolo che ha le sue dimensioni. */
public class AppletConParametri extends JApplet {
    String desc;    // descrizione dell'applet
    int larg, alt;    // larghezza e altezza dell'applet
    /* Inizializza l'applet. */
    public void init() {
        this.desc = this.getParameter("descrizione");
        this.larg = this.getWidth();
        this.alt = this.getHeight();
    }
    /* Visualizza la descrizione dell'applet. */
    public void paint(Graphics g) {
        g.drawRect(0, 0, this.larg-1, this.alt-1);
        g.drawString(descrizione, this.larg/2, this.alt/2);
    }
}
```

Applet con parametri

Visualizzazione della pagina contenente i due applet **AppletConParametri**



Applet e variabili d'istanza

Un applet è un oggetto istanza

- come tale, può avere uno stato (descritto dalle sue variabili d'istanza) e un comportamento (descritto dai suoi metodi d'istanza)
- nei casi più semplici, le variabili d'istanza di un applet vengono utilizzate per rappresentare informazioni condivise tra i vari metodi dell'applet
  - è il caso delle variabili **descrizione**, **larg** e **alt** dell'esempio precedente, che sono inizializzate dal metodo **init()** e accedute dal metodo **paint(…)**
  - in pratica, spesso un applet modella un oggetto, ma anche il piccolo programma che lo gestisce
- nei casi più complessi, un applet modella semplicemente il piccolo programma immerso in pagina web che gestisce un insieme di oggetti software, ciascuno con il suo stato e il suo comportamento