

# Corso sul linguaggio Java

## Modulo JAVA8

### B1 – Accesso sequenziale

M. Malatesta B1 - Accesso sequenziale-18

1  
11/11/2012

## Prerequisiti

- Programmazione ad oggetti
- Conoscenza classi di base di I/O
- Tecnica della programmazione

M. Malatesta B1 - Accesso sequenziale-18

2  
11/11/2012

# Introduzione

In questa Unità vediamo come si progetta un'applicazione che gestisce dati su un file organizzato in modo sequenziale, facendo uso dell'accesso sequenziale.

M. Malatesta B1 - Accesso sequenziale-18

3  
11/11/2012

# L'applicazione

Realizziamo una semplice applicazione per gestire libri. Occorre creare:

- una classe *Libro.java* che presenta gli attributi tipici (codice, autore, editore, titolo e prezzo) e metodi di default;
- una classe *Libreria.java* che implementa le tipiche operazioni utente che per semplicità riduciamo a:
  - inserimento dei dati di un libro;
  - stampa di tutti i record presenti nell'archivio;
  - visualizzazione della scheda di un libro;
  - eliminazione dei dati di un libro.

M. Malatesta B1 - Accesso sequenziale-18

4  
11/11/2012

## La classe *Libro.java*

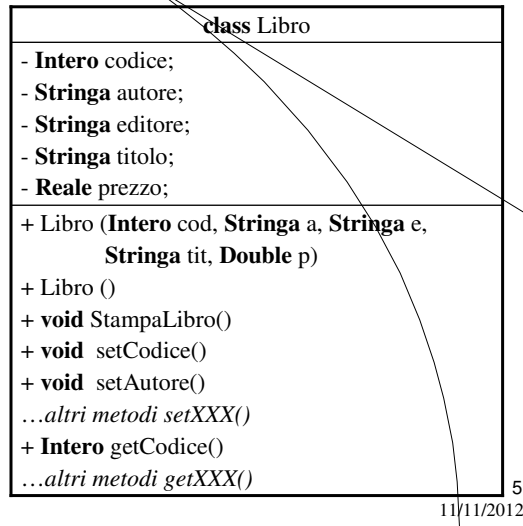
**ATTIVITA':** rappresentare la classe *Libro* in UML.

La classe deve prevedere i costruttori, i metodi *setXXX()* e *getXXX()* per gli attributi ed un metodo di stampa globale.

Questa classe viene progettata secondo la UML mostrata a fianco.

Una volta creata la classe, si testa mediante una semplice applicazione *TestLibro.java*.

M. Malatesta B1 - Accesso sequenziale-18



11/11/2012 5

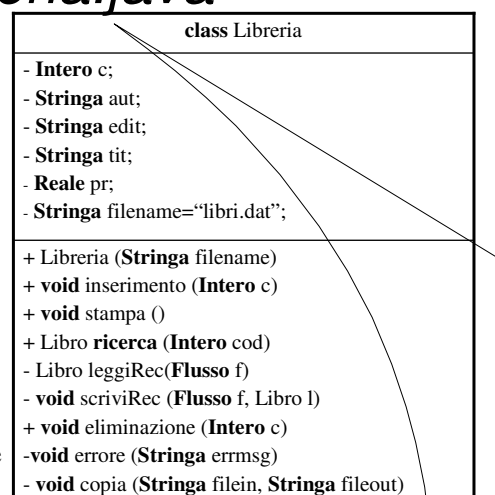
## La classe *Libreria.java*

**ATTIVITA':** rappresentare la classe *Libreria* in UML.

Oltre ai metodi pubblici indicati, è bene inserire due *metodi di servizio privati* che servono a semplificare la struttura dei metodi pubblici:

- *Libro leggiRec(Flusso f)* che restituisce il record letto dal file;
- *scriviRec (Flusso f, Libro l)* che scrive l'oggetto *l* nel flusso *f*.

M. Malatesta B1 - Accesso sequenziale-18



11/11/2012 6

# Specifiche dell'applicazione

Facciamo sull'applicazione *Libreria.java* le seguenti osservazioni:

- il file di dati *filename* è sequenziale e le aggiunte vengono fatte in coda, per cui si suppone non ordinato;
- la funzione di eliminazione richiede il file ausiliario e la successiva copia, per cui viene introdotto il metodo privato  
**void** copia (**Stringa** filein, **Stringa** fileout)
- l'applicazione prevede un menu operativo, in cui l'utente sceglie con una lettera l'operazione desiderata;
- i dati nel file vengono gestiti come dati primitivi;

M. Malatesta B1 - Accesso sequenziale-18

7  
11/11/2012

# Specifiche dell'applicazione

- l'applicazione fa uso di stream binario per l'input
  - **FileInputStream** come *stream* di input
  - **DataInputStream** per gli *stream* di dati primitivi
- l'applicazione fa uso di stream binario per l'output :
  - **FileOutputStream** come *stream* di output
  - **DataOutputStream** per gli *stream* di output
- un metodo per stampare i messaggi di errore
  - **void** errore (**Stringa** errmsg)
- i metodi *inserimento(...)* ed *eliminazione(...)* usano come parametro il codice di un libro, la cui presenza viene preventivamente testata nel main, tramite il metodo
  - + Libro **ricerca** (**Intero** cod)

M. Malatesta B1 - Accesso sequenziale-18

8  
11/11/2012

# Inserimento

```
public void inserimento (int c)
{
    Libro l;
    variabili locali;
    try {FileOutputStream fout =new FileOutputStream (filename, true);
        DataOutputStream dataout = new DataOutputStream (fout);
        aut = JOptionPane.showInputDialog("Autore: ");
        lettura altri campi
        l = new Libro(c, aut, edit, tit, pr);
        scriviRec (dataout, l);
        dataout.flush();
        dataout.close();
    }
    catch (IOException e)
    { errore("Errore in inserimento!");
```

Viene  
descritto  
in seguito

M. Malatesta B1 - Accesso sequenziale-18

9  
11/11/2012

# Stampa

```
public void stampa() throws IOException
{
    Libro l;
    int ret;
    try { FileInputStream fin= new FileInputStream (filename);
        DataInputStream datain = new DataInputStream (fin);
        while ((l = leggiRec(datain))!=null)
            l.stampaLibro();
        datain.close();
    }
    catch (IOException e)
    { errore ("File vuoto");
```

Viene  
descritta  
in seguito

M. Malatesta B1 - Accesso sequenziale-18

10  
11/11/2012

# Ricerca

```
public Libro ricerca (int cod)      // ricerca sequenziale
{
    Libro l;
    try
    {
        FileInputStream fin = new FileInputStream (filename);
        DataInputStream datain = new DataInputStream (fin);
        while ((l = leggiRec(datain)) != null && l.getCodice() != cod);
        if (c == cod) return l;
        else { datain.close();
               return null;
            }
    }
    catch (IOException e)
    {
        return null;
    }
}
```

Viene  
descritta  
in seguito

M. Malatesta B1 - Accesso sequenziale-18

11  
11/11/2012

# Lettura

```
private Libro leggiRec (DataInputStream dis)
{
    try
    {
        c=dis.readInt();
        aut=dis.readUTF();
        edit=dis.readUTF();
        tit=dis.readUTF();
        pr=dis.readDouble();
        Libro l = new Libro(c, aut, edit, tit, pr);
        return l;
    }
    catch (IOException e)
    {
        errore ("Fine file");
        return null;
    }
}
```

M. Malatesta B1 - Accesso sequenziale-18

12  
11/11/2012

# Scrittura

```
private void scriviRec (DataOutputStream dos, Libro l)
{ try
    { dos.writeInt(l.getCodice());
      dos.writeUTF(l.getAutore());
      dos.writeUTF(l.getEditore());
      dos.writeUTF(l.getTitolo());
      dos.writeDouble(l.getPrezzo());
    }
    catch (IOException e)
    { errore ("Errore in scrittura"); }
}
```

M. Malatesta B1 - Accesso sequenziale-18

13  
11/11/2012

# Eliminazione

```
public void eliminazione (int c) throws IOException
{ Libro l=null;
  FileInputStream fin = new FileInputStream (filename);
  DataInputStream in = new DataInputStream (fin);
  FileOutputStream foux =new FileOutputStream ("temp.dat");
  DataOutputStream outx = new DataOutputStream (foux);
  while((l = leggiRec(in))!=null)
    if (l.getCodice()!=c) scriviRec(outx, l);
  in.close();
  outx.close();
  copia("temp.dat", filename);
}
```

Viene descritta in seguito

M. Malatesta B1 - Accesso sequenziale-18

14  
11/11/2012

# Copia

```
private void copia (String filein, String fileout) throws IOException
{
    Libro l;
    FileInputStream finx =new FileInputStream (filein);
    DataInputStream inx = new DataInputStream (finx);
    FileOutputStream fout =new FileOutputStream (fileout);
    DataOutputStream out = new DataOutputStream (fout);
    while((l=LeggiRec(inx))!=null)
        ScriviRec(out, l);
    inx.close();
    out.close();
}
```

**ATTIVITA’:** implementare il metodo *modifica (...)* partendo dallo schema di *eliminazione()*

M. Malatesta B1 - Accesso sequenziale-18

15  
11/11/2012

# Argomenti

- L'applicazione
- La classe *Libro.java*
- La classe *Libreria.java*
- Specifiche dell'applicazione
- Inserimento
- Stampa
- Ricerca
- Lettura
- Scrittura
- Eliminazione
- Copia

M. Malatesta B1 - Accesso sequenziale-18

16  
11/11/2012



# Altre fonti di informazione

- P.Gallo, F.Salerno – Java, ed. Minerva Italica
- M. Bigatti – Il linguaggio Java, ed. Hoepli

M. Malatesta B1 - Accesso sequenziale-18

17  
11/11/2012