Programmazione Orientata agli Oggetti in Linguaggio Java

Programmazione su XML: Conclusioni

versione 2.0

G. Mecca – Università della Basilicata – mecca@unibas.it

Sommario

- Ricapitolazione
- Diagrammi UML
- API di Java
  - ⇨Copia di File
  - ⇨Riferimenti ai File

Ricapitolazione

- XML
  - ⇨una tecnologia recente, ma ormai ampiamente utilizzata in vari contesti
- Perché bisogna conoscerla
  - ⇨per poterla utilizzare quando le applicazioni lo richiedono (es: file di configurazione, persistenza ecc.)
  - ⇨per poter sviluppare nuove applicazioni basate su XML

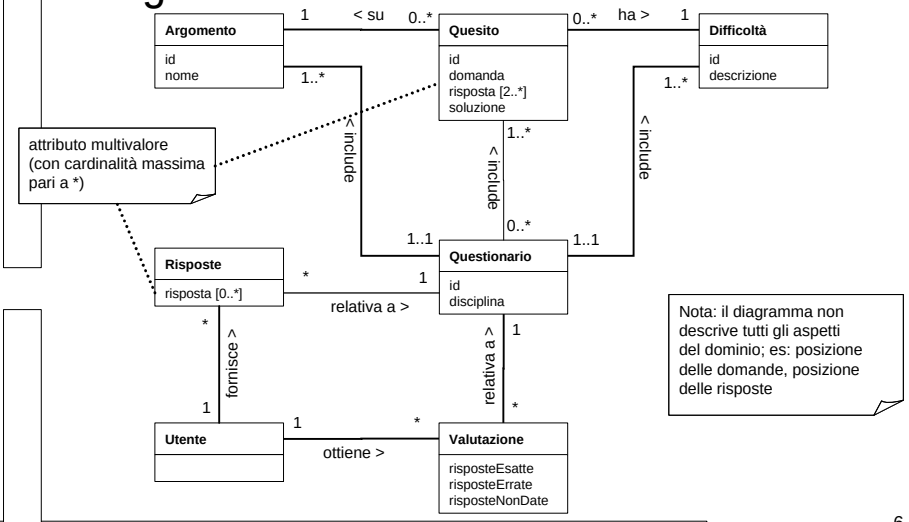
Ricapitolazione

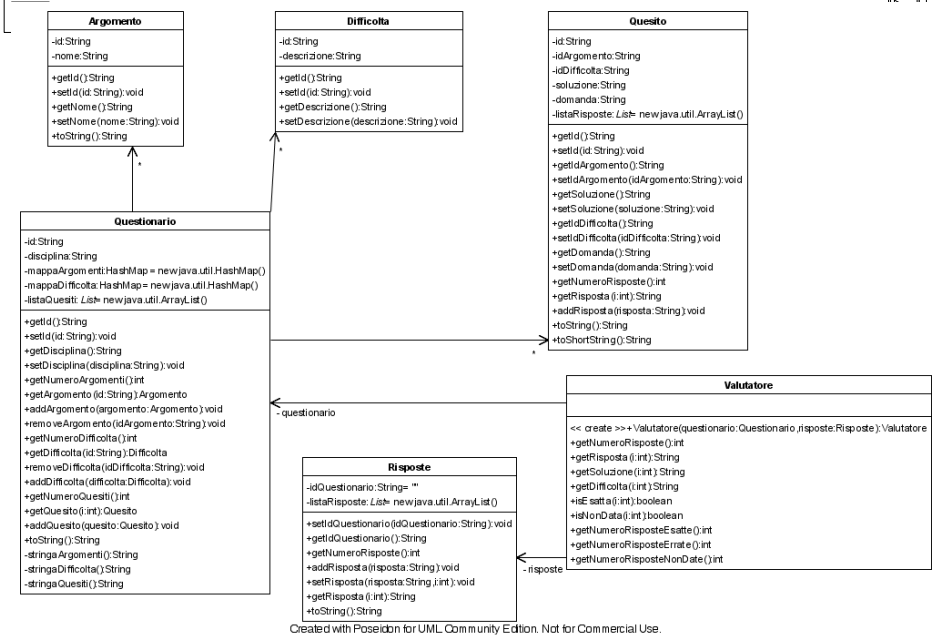
- In questa unità
  - ⇨discutiamo alcuni aspetti conclusivi relativi ai progetti di riferimento
  - ⇨utilizzo delle API e degli strumenti di Java
- In particolare
  - ⇨utilizzo di file jar per la distribuzione
  - ⇨copia dei file

Diagrammi UML

- L'applicazione questionari
  - ⇨vediamo per concludere i diagrammi UML
- Al solito
  - ⇨prima il diagramma concettuale
  - ⇨poi il diagramma finale delle classi
- Nota
  - ⇨in questo caso il diagramma delle classi è troppo complesso per essere mostrato tutto; ci concentriamo solo sul modello

Diagrammi UML

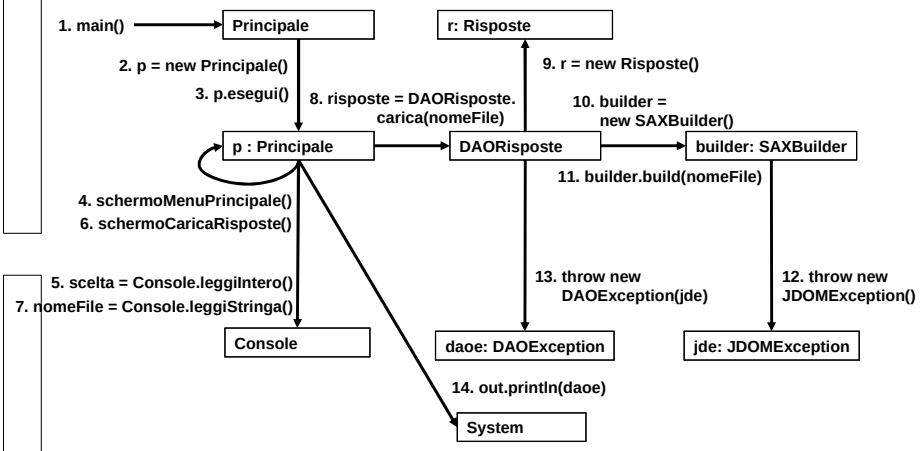


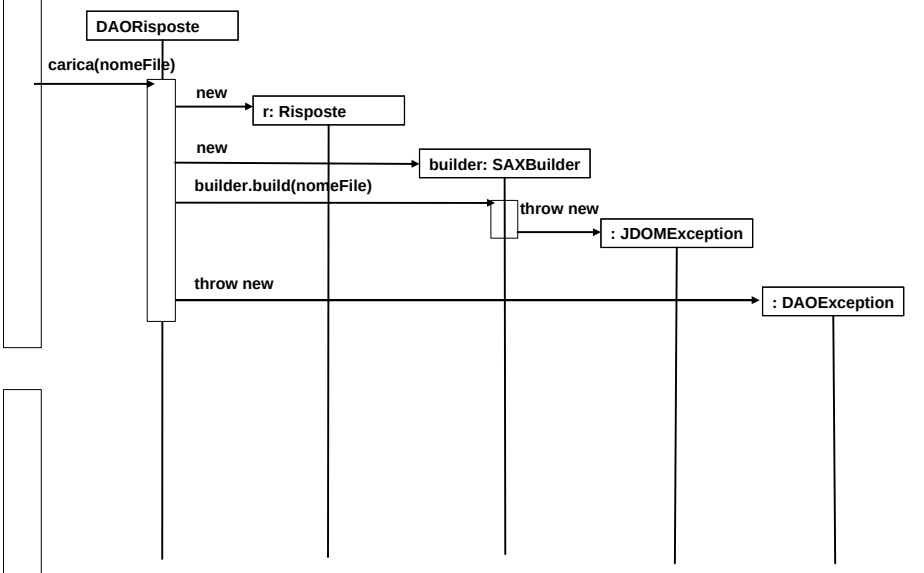


Diagrammi UML

- Diagrammi dinamici
  - ⇨utilizziamo i diagrammi dinamici per ridiscutere la tecnica di trasformazione delle eccezioni vista nell'unità precedente
  - ⇨vedremo prima il diagramma di collaborazione
  - ⇨e poi il diagramma di sequenza (per ragioni di spazio viene mostrato solo un frammento)

Diagrammi UML





API di Java

- Un aspetto interessante dell'applicazione
  - ⇨il salvataggio delle risposte
  - ⇨richiede, infatti, il salvataggio del DTD nella stessa cartella in cui viene salvato il file delle risposte
  - ⇨altrimenti, al successivo caricamento, il DTD non sarebbe disponibile e la convalida fallirebbe

Copia di File

- La copia di file in Java
  - ⇨non è un'operazione scontata
  - ⇨non esiste un metodo "copia"
  - ⇨bisogna effettuarla byte a byte
- In particolare
  - ⇨è necessario creare un flusso per ciascuno dei due file
  - ⇨e poi copiare i byte da un flusso all'altro

```
package it.unibas.questionari.persistenza;

public class DAOUtilita {

private static final String PERCORSODTD = "e:\\codice\\";
private static final String NOMEDTD = "risposte.dtd";

public static void copiaDTD(String percorso) throws DAOException {
      try {
         String nomeDTDOriginale = PERCORSODTD + NOMEDTD;
         java.io.File fileDTDOriginale = new java.io.File(nomeDTDOriginale);
         String nomeNuovoDTD = percorso + NOMEDTD;
         java.io.File nuovoFileDTD = new java.io.File(nomeNuovoDTD);
         copiaFile(fileDTDOriginale, nuovoFileDTD);
      } catch (java.io.IOException ioe) {
         System.err.println(ioe);
         throw new DAOException(ioe);
      }
   }
```



```
private static void copiaFile(java.io.File source, java.io.File dest)
                                    throws java.io.IOException {
      java.io.FileInputStream in = null;
      java.io.FileOutputStream out = null;
      try {
         in = new java.io.FileInputStream(source);
         out = new java.io.FileOutputStream(dest);
         byte[] buffer = new byte[1024];
         int byteLetti;
         while ((byteLetti = in.read(buffer)) > 0) {
            out.write(buffer, 0, byteLetti);
         }
      } finally {
         if (out != null) {out.close();}
         if (in != null) {in.close();}
      }
   }
```

viene usato un array di byte

il metodo read() legge dal flusso 1024 byte alla volta e restituisce il numero di byte letti; -1 nel caso il file sia finito

il metodo write() ha tre argomenti: l'array di byte, l'indice del I elemento da copiare, l'indice dell'ultimo elemento da copiare

Riferimenti ai File

- Attenzione
  - ⇨nel metodo di copia bisogna specificare la posizione del file originale
- La soluzione utilizzata
  - ⇨riferimento assoluto (e:\codice\risposte.dtd)
  - ⇨ma questo provoca vari problemi nel caso in cui l'applicazione sia installata su macchine diverse da quella di sviluppo

Riferimenti ai File

- Una soluzione molto migliore
  - ⇨specificare un riferimento relativo alla posizione della classe in cui viene utilizzato
  - ⇨es: .\risposte.dtd oppure ..\varie\risposte.dtd
- Ma, in Java...
  - ⇨non c'è un modo semplice per farlo
  - ⇨questo dipende dal meccanismo di risoluzione basato sul classpath (le classi possono essere eseguite da diverse posiz.)

Riferimenti ai File

- In Java
  - ⇨è possibile utilizzare riferimenti relativi ai file
  - ⇨è possibile chiedere alla macchina virtuale di espandere un riferimento relativo in un riferimento assoluto
  - ⇨metodo getAbsolutePath() di java.io.File
  - ⇨ma il riferimento viene espanso utilizzando come cartella base la cartella personale dell'utente (proprietà di sistema user.dir)

Riferimenti ai File

- Le possibili soluzioni
  - ⇨utilizzare posizioni "notevoli" del disco
  - ⇨oppure utilizzare il CLASSPATH
- Posizioni notevoli del disco
  - ⇨per esempio la cartella dei file temporanei oppure la cartella dei file dell'utente
  - ⇨corrispondono ai valori delle proprietà di sistema java.io.tmpdir e user.dir

Riferimenti ai File

- Esempio: IndovinaIlNumero
  - ⇨scrittura del record

```
package it.unibas.indovinailnumero;
public class Record {

    private String nomeFileRecord;

    public Record() {
        String tempDir = System.getProperty("java.io.tmpdir");
        this.nomeFileRecord = tempDir + "recordIndovina.txt";
    }
```


Riferimenti ai File

- Utilizzare il classpath
  - ⇨una soluzione complessa
  - ⇨è possibile chiedere al ClassLoader di localizzare un file nelle cartelle visibili lungo il classpath
  - ⇨attraverso il metodo getResourceAsStream() della classe java.lang.Class
  - ⇨restituisce uno stream da cui è possibile leggere il contenuto di un file

```
public final static String DTD = “risposte.dtd”;

private void copiaDtdEStile(String percorso) throws DAOException {
      java.io.InputStream in = null;
      java.io.OutputStream out = null;
      try {
         java.lang.Class classe = DAOUtilita.class;
         in = classe.getResourceAsStream(DTD);
         out = new FileInputStream(percorso + “\\” + DTD);
         byte[] buffer = new byte[1024];
         int byteLetti;
         while ((byteLetti = in.read(buffer)) > 0) {
            out.write(buffer, 0, byteLetti);
         }
      } catch (IOException ioe) {
         throw new DAOException(ioe);
      } finally {
         if (out != null) {out.close();}
         if (in != null) {in.close();}
      }
}
```

viene cercato un file di nome risposte.dtd nelle cartelle e nei jar del classpath e viene restituito il riferimento ad uno stream per leggerne il contenuto

Riassumendo

- Ricapitolazione
- Diagrammi UML
- API di Java
  - ⇨Copia di File
  - ⇨Riferimenti ai File

Termini della Licenza