Belloni Francesco Matricola 725676

Facedoor

22 Agosto 2015

1 Strutture dati utilizzate

Il progetto è una versione semplificata di un social network che non utilizza nessun database ma ha tutti i dati dei vari utenti iscritti in un unico file csv. Per accedere ai dati con maggiore semplicità, si è deciso di implementare la gestione degli utenti con una lista. La struttura è popolata da classi "utenti" che racchiudono i vari metodi per accedere ai campi (es. nome, cognome, ID,...).

Per tutte le varie operazioni che servono per accedere al file csv, è stata usata la libreria Apache Commons CSV. Per velocizzare la gestione del file csv e quindi la realizzazione del progetto, si è preso ispirazione da questo articolo, modificandolo in base alle specifiche del compito assegnato: http://examples.javacodegeeks.com/core-java/apache/commons/csv-commons/writeread-csv-files-with-apache-commons-csv-example/ di Ashraf Sarhan

La classe utenti ("student" nel progetto) che rappresenta l'utente inscritto a Facedoor, è una semplice classe contenitore che racchiude i vari dati anagrafici (es. nome-cognome) e tutte le informazioni aggiuntive (es. foto-cv); ha un costruttore e necessita di tutti i dati dell'utente. Questi ultimi vengono letti dal file csv quando il programma viene lanciato oppure in fase di registrazione quando viene creato un nuovo utente.

2 Metodi/algoritmi di gestione delle strutture dati

Il programma si presenta senza interfaccia grafica ma tramite linea di comando. Le operazioni di output sono quindi lo stampare a video i vari menù di opzioni tra cui l'utente può scegliere e i risultati trovati in Facedoor.

I vari menù principali sono stati numerati e una volta che ci si sposta nel menù successivo viene semplicemente settato `id_menu` (una variabile globale che

rappresenta il menù in cui siamo attualmente) e richiamata la funzione di *ESEGUI_MENU()*. La funzione semplicemente con un costrutto **switch** decide quale menù stampare a video. I vari menù sono:

```
// SCELTE MENU
// 0 Menu principale - 1 Registrati - 2 Login - 3 Logout - 4 Modifica dati
// 5 Ricerca - 6 Esci
```

Nell'inserimento dei dati dell'utente, dove è risultato possibile, è stato effettuato un controllo (es. formato email valido, opzione giusta tra profilo pubblico/privato , userID unico)

Per semplicità il programma ha delle variabili globali che ci permettono di accedere alle informazioni "sensibili", visto che viene eseguito solo in locale e non contemporaneamente su più dispositivi. Le variabili più significative sono:

```
static int id_menu=0; // Mi indica in che menù/sottomenù sono

public static String NextIDUser; // L'ID del nuovo utente

public static List<Student> List_students; // Lista degli utenti registrati
```

Per navigare nei profili dei vari utenti viene usato un indice numerico unico per ogni iscritto: ID. Il numero ID, per mantenere la sua unicità, viene salvato su di un file esterno e si incrementa ad ogni nuova registrazione (*NextIDUser*).

Per gestire la lista degli utenti, non essendo numerosi, si è scelto di iterare una lista fino a quando non si trovano le informazioni necessarie.

3 Altri metodi

Per quanto riguarda invece la ricerca, si è utilizzata la classe StringSimilarity (http://stackoverflow.com/questions/955110/similarity-string-comparison-in-java)

4 Complessità

L'algoritmo usato per la funzione cerca è composto da due cicli "for" innestati: il primo fa scorrere i caratteri della stringa più lunga mentre il secondo fa scorrere i caratteri della parola più corta. [costo O(n^2)]