Progetto Rubrica

Realizzare un progetto in Java che rappresenti una **rubrica telefonica**, un software che gestisca i contatti.

Il software deve avere una classe **Persona** che contenga le informazioni di uno dei contatti della rubrica. Lo stesso software deve contenere una **lista** di oggetti (ad esempio con la classe Vector) di tipo persona. Deve permettere la creazione, la modifica e l'eliminazione delle persone esistenti.

La persona deve mantenere le seguenti informazioni:

- nome: stringa
- cognome: stringa
- indirizzo: stringa
- telefono: stringa
- eta: intero

Interfaccia grafica

L'interfaccia dell'applicazione deve essere realizzata con la libreria java base SWING. Ci deve essere una **finestra principale**, che mostra una JTable con una riga per ogni persona. Le colonne devono mostrare solo nome, cognome e telefono. In basso ci devono essere tre bottoni:

- nuovo: serve per creare una nuova persona
- modifica: serve per modificare una persona esistente
- elimina: serve per eliminare una persona esistente

I tasti nuovo e modifica devono portare all'apertura di una seconda finestra, chiamata **editor-persona**, che serve per inserire e modificare i dati di una persona. Questa finestra deve avere una serie di campi, divisi per riga, nomeCampo – valoreCampo, per ogni dato della persona (usando JLabel e JTextField). Questa finestra è sostanzialmente un modulo di inserimento dati per un elemento Persona. Questa finestra deve avere due bottoni: salva e annulla.

Funzionamento bottoni

I flussi di lavoro dei bottoni sopra citati sono i seguenti:

Finestra principale, bottone nuovo: quando si clicca sul bottone nuovo, sopra la finestra principale deve apparire la finestra editor-persona, con tutti i campi vuoti.

Finestra principale, bottone modifica: questo bottone serve per la modifica di una persona. Se si clicca questo bottone, senza aver selezionato prima una riga della tabella, dovrà essere mostrato un errore (JOptionPane.showMessageDialog(…) che ci informa che per modificare è necessario

prima selezionare una persona. Se si seleziona prima una persona e poi si preme modifica, sopra la finestra principale deve apparire la finestra editor-persona, con tutti i campi valorizzati ai relativi attributi della persona selezionata.

Finestra principale, bottone elimina: questo bottone serve per l'eliminazione di una persona. Se si clicca questo bottone, senza aver selezionato prima una riga della tabella, dovrà essere mostrato un errore (JOptionPane.showMessageDialog(…) che ci informa che per eliminare è necessario prima selezionare una persona. Se si seleziona prima una persona e poi si preme elimina, sopra la finestra principale deve apparire un messaggio di conferma (JOptionPane.showConfirmDialog(…)) che chiede all'utente: "Eliminare la persona NOME COGNOME?". Se si preme no, non deve accadere nulla, solo chiudere la finestra di conferma. Se si preme si, oltre a chiudere la finestra di conferma, bisogna eliminare la persona dalla lista di persone, aggiornare la tabella in modo che non la visualizzi più.

Finestra editor persona, bottone salva: questo bottone salva i dati della persona, visualizzati in finestra. Il salvataggio modifica i dati della persona e chiude la finestra editor-persona.

Finestra editor persona, bottone annulla: questo bottone chiude la finestra editor-persona senza salvare i dati della persona.

Persistenza

Dopo aver realizzato la parte logica e la parte di interfacce grafiche, totalmente funzionanti, bisogna implementare lo strato di persistenza della nostra applicazione. In assenza dello strato di persistenza, il nostro programma gestirebbe correttamente la gestione delle persone. Permetterebbe inserimenti e cancellazioni, ma allo spegnimento e riavvio, mostrerebbe sempre la rubrica vuota. Sarebbe dunque privo di memoria.

Procederemo alla realizzazione della persistenza mediante salvataggio su file. Tramite la classe Scanner leggeremo da file e tramite la classe PrintStream scriveremo su file. Useremo la classe File per identificare il file. Nel file inseriremo il contenuto dei dati delle persone seguendo la seguente codifica:

informazioni.txt ----------

```
Steve;Jobs;via Cupertino 13;0612344;56

Bill;Gates;via Redmond 10;06688989;60

Babbo;Natale;via del Polo Nord;00000111;99
```

Il programma, all'avvio, dovrà caricare in maniera invisibile per l'utente, tutto il contenuto informativo del file informazioni.txt (se il file non esiste, non avverrà nulla e non ci saranno errori).

Se il file è presente, ogni riga di dati separati da punto e virgola sarà caricata dentro un oggetto persona e visualizzato nella tabella.

Inoltre, ad ogni salvataggio di persona, nella finestra editor-persona, alcune classi dello strato logico si preoccuperanno, dopo aver aggiunto la persona nella lista, di salvare tutte i dati di tutte le persone della lista nel file con la stessa codifica.

Si consiglia l'utilizzo di un programma IDE per lo sviluppo come Eclipse.

Il programma rubrica dovrà essere esportato nel formato JAR come "runnable jar" e con il nome Rubrica.jar. Questo permetterà di eseguire il programma facendo semplicemente doppio-click, con una macchina dove è installata e configurata una JVM.

Test

Prima di inviare il pacchetto Rubrica.jar, testarlo a fondo in ogni sua funzione. Provare i seguenti casi di test:

1) Apro programma, aggiungo persona, verifico la presenza in tabella
2) Apro programma, aggiungo persona, clicco su persona, faccio modifica, verifico che i dati escano in finestra editor-persona
3) Apro programma, aggiungo 2-3 persone. Elimino la persona di mezzo. Modifico l'ultima della lista. Verifico che i dati in tabella siano coerenti con le eliminazioni e modifiche. Apro tutte le persone, verifico che i dati siano coerenti.
4) Apro programma, aggiungo 1 persona, lo chiudo. Apro programma, verifico che la persona sia correttamente mantenuta sia in tabella che aprendo i dati in editor-persona. Aggiungo altre 2 persone. Richiudo e riapro, ri-verifico se c'è tutto e non ci sono errori.
5) Verifico tra un'apertura e la successiva che sul file informazioni.txt il contenuto sia corretto.
6) Verificare il funzionamento del programma anche su un computer diverso.

Evoluzioni EXTRA

Chi terminasse rapidamente il progetto, e fosse interessato ad aggiungere funzionalità e valore a questo piccolo programmino, può trarre spunto dai seguenti suggerimenti, per inserire nuove caratteristiche. Non è necessario seguire l'ordine, si possono aggiungere funzionalità come si vuole e anche sulla base delle proprie competenze e piacere.

1) Modificare il salvataggio dati: non salvare su un singolo file "informazioni.txt", ma salvare i dati, all'interno di una cartella "informazioni", creando un file per ogni persona. E' possibile salvare creando vari file con ordine progressivo "Persona1.txt", "Persona2.txt", etc… Oppure è possibile creare un file con nome "NOME-COGNOME.txt" per ogni persona. Attenzione a persone con uguale nome e cognome.

2) Creare una seconda classe di “dominio” Utente. Oltre alle Persona, avremo anche gli Utente, che contengono dati relativi all’utente che accede al software. Un utente dovrà avere un username e una passoword. Poi bisognerà creare una nuova piccola finestra per il login, che contiene i campi Utente e Passoword e il bottone LOGIN. Quando si avvia l’applicazione, l’unica finestra che dovrà essere mostrata sarà il login e non la finestra principale. Se un utente inserisce i dati di login corretti, la finestra di login si chiuderà e si aprirà la finestra principale, dando all’utente la libertà di usare il software. Se i dati di login non sono corretti, bisognerà mostrare un messaggio di “login errato” e non mostrare la finestra principale.
3) Aggiungere una barra degli strumenti (JToolbar) in ogni finestra, e caricare i bottoni all’interno della barra, piuttosto che visualizzarli in basso. E’ possibile anche caricare delle piccole immagini nei bottoni della toolbar per abbellirne l’aspetto.
4) Caricare il codice sorgente del programma su “sistemi di versioning online” come GitHub o BitBucket. Questo permetterebbe di rendere pubblico il proprio codice (chiunque potrà leggerlo e usarlo). Sarebbe uno sviluppo di un progetto open-source ed è possibile condividerlo anche con altri sviluppatori.
5) Salvataggio dati su **database**. Volendo sostituire lo strato di classi per la persistenza, in modo che non salvino su file ma su database, è possibile installare il software per la gestione di database gratuito e open-source MySql. Dopo averlo installato si dovrà creare un database e delle tabelle, una per gestire le persone e una per gestire gli utenti. Dopo si dovrà sfruttare le classi e le interfacce della sezione JDBC (che fanno parte della struttura di classi di java base) per accedere al db mysql dal nostro software e memorizzare i dati nel db. Per sfruttare JDBC e collegarsi a MySQL, sarà necessario scaricare una piccola libreria java in formato .jar chiamata connector/J di MySQL. Questa libreria dovrà essere aggiunta alle librerie-dipendenze del nostro progetto in eclipse. Per permettere il test di questa applicazione presso un computer di terzi è necessario inviare a corredo del pacchetto applicativo **Rubrica.jar** ulteriori informazioni relative al database da replicare presso il computer di chi lo testerà. Nello specifico, considerando che sul computer di chi lo testa sarà già istallato un sistema MySQL, sarà necessario replicare lo stesso identico database (stesso nome, stesse tabelle etc.). Per questo, si richiede di aggiungere “al pacchetto da inviare” un file **schema_database.sql** che sarà un file di testo contenente tutte le istruzioni SQL da lanciare per la costruzione del database stesso. Possiamo considerare che il lancio di questo script sia parte dell’istallazione dell’applicazione rubrica. Inoltre, poiché ogni sistema MySQL possiede le proprie credenziali (username, password) di accesso, differenti e spesso immodificabili, è necessario “parametrizzare” questi dati esternamente all’applicazione **Rubrica.jar.** Il consiglio più comodo e semplice è quello di creare un terzo file da rilasciare, in formato .properties, facilmente leggibile con Java, nel quale l’utente finale durante l’istallazione, inserirà i seguenti dati: username, password, ip-server-mysql, porta. Dunque il pacchetto finale da inviare come progettino rubrica sarà un file zippato contenente:
    - **Rubrica.jar**
    - **schema_database.sql**

- **credenziali_database.properties**

L'utente che dovrà provare l'applicazione dovrà in sequenza:

1. decomprimere l'archivio .zip
2. aprire e sostituire le credenziali del proprio sistema MySQL dentro credenziali_database.properties
3. lanciare sul proprio sistema MySQL lo script di costruzione database schema_database.sql
4. eseguire Rubrica.jar

Una nota sull'invio del progetto: poiché le caselle di posta elettronica bloccano file con estensione .zip o simili, è consigliabile rinominare l'archivio zip eliminando l'estensione per il trasferimento. Oppure sfruttare un sistema come WeTransfer o Dropbox.