



UNIVERSITÀ DEGLI STUDI DI PADOVA

*Relazione progetto didattico LinQedIn - Programmazione ad Oggetti
AA 2014 / 2015*

LinQedIn

Nicola Carraro mat. 1002050

INDICE

1 Specifica del progetto	2
2 Sviluppo del progetto	3
2.1 Parte logica	3
2.1.1 Classe DatabaseLinQedIn	3
2.1.2 Classe Utente	4
2.1.3 Classe IdUtente	4
2.1.4 Classe Profilo	4
2.1.5 Classe ReteSociale	4
2.1.6 Classe IODatabase	4
2.1.7 Classe LinQedInAmministratore	4
2.1.8 Classe LinQedInUtente	5
2.2 Parte grafica	5
2.2.1 Classe QtFinestraPrincipale	5
2.2.2 Classe QtFinestraAmministratore	5
2.2.3 Classe QtFinestraUtente	5
2.2.4 Classe QtFinestraUtenteVisualizzazione	6
2.3 Funzionamento	6
3 Note progettuali	7

CAPITOLO 1

SPECIFICA DEL PROGETTO

Lo scopo del progetto è lo sviluppo in C++/Qt di un sistema minimale per l'amministrazione ed utilizzo tramite interfaccia utente grafica di un (piccolo) database di contatti professionali ispirato a LinkedIn. LinkedIn è il principale servizio web di rete sociale per contatti professionali, gratuito ma con servizi opzionali a pagamento.

CAPITOLO 2

Sviluppo del progetto

Il programma LinQedIn, come richiesto dalla specifica, si sviluppa principalmente in due parti:

- Parte logica, nella quale viene definito il database degli utenti del sistema, le classi e le funzioni necessarie ad operare su di esso e a gestire gli utenti, i metodi per la scrittura e la lettura su file dei dati
- Parte grafica, nella quale vengono definite le classi per la gestione del sistema tramite interfacce grafiche opportune

2.1 Parte logica

La parte logica si occupa della definizione del database degli utenti, cuore del programma. Esso permette di gestire una gerarchia di classi dedicata agli utenti e alle loro diverse tipologie, e di effettuare operazioni di inserimento, rimozione e modifica su di essi.

In seguito vengono descritte le principali classi e la loro funzione.

2.1.1 Classe DatabaseLinQedIn

È la classe che si occupa della gestione del database dal punto di vista logico. Creato attraverso una lista di puntatori smart per ottimizzare la gestione della memoria condivisa, contiene dei puntatori ad oggetti della classe base astratta *Utente*, così da poter gestire tutte le classi derivate da questa e quindi le diverse tipologie di utente. *DatabaseLinQedIn* contiene la classe *SmartP* per la gestione della memoria e la classe *DatabaseItem* che rappresenta un oggetto del database e contiene il puntatore alla classe Utente. Mette inoltre a disposizione una classe *Iteratore* per poter scorrere gli elementi del database anche da classi esterne. Fornisce le funzioni di inserimento, rimozione e ricerca di un utente.

2.1.2 Classe Utente

La classe *Utente* è la classe base astratta per la definizione di un oggetto Utente. Un Utente è composto da un *IdUtente*, da un *Profilo* e da una *ReteSociale*. Queste componenti sono tre campi dati della classe Utente e verranno descritti in seguito. All'interno di essa è presente la classe interna *RicercaFuntore* e la ridefinizione dell'operatore funzione, per permettere la ricerca da parte di un utente in modo diversificato rispetto alla sua tipologia, come richiesto dalla specifica di progetto. Ciò è permesso attraverso l'utilizzo dei funtori e della funzione *ricerca*, una funzione virtuale pura nella classe *Utente* e ridefinita quindi nelle sue sottoclassi, che chiama il metodo opportuno a seconda del tipo di Utente. Le classi derivate semplicemente implementano la funzione *ricerca* e forniscono un metodo per restituire il tipo dell'Utente.

2.1.3 Classe IdUtente

È la classe alla quale appartiene il primo campo dati di un oggetto della classe *Utente*. Rappresenta l'identificato dell'utente, nonché il suo nome utente, ed è composto da un campo dati di tipo *string* formato dall'indirizzo email dell'utente stesso.

2.1.4 Classe Profilo

È la classe più corposa contenente le informazioni dell'Utente. Composta da diversi campi dati quali *Nome*, *Cognome*, ecc. fornisce anche i rispettivi metodi di modifica di essi. È composta inoltre da altri campi dati contenenti informazioni aggiuntive e non obbligatorie. Queste informazioni sono oggetti di classi esterne, come ad esempio *TitoloDiStudio*, *Lingue*, e sono contenuti in apposti array. La classe *Profilo* mette a disposizione anche i metodi per la gestione di queste informazioni aggiuntive, quindi l'aggiunta e la rimozione di esse dal profilo dell'Utente.

2.1.5 Classe ReteSociale

Questa classe si occupa della gestione della rete di utente che un Utente può avere, fornendo i metodi di aggiunta e rimozione di essi.

2.1.6 Classe IODatabase

La classe *IODatabase* si occupa della lettura e della scrittura dei dati su file. Utilizza le opportune classi *QXmlStreamReader* e *QXmlStreamWriter* per importare ed esportare i dati su file .xml esterni. Per facilitare la scrittura su file, in ogni classe riguardante l'Utente e le sue informazioni, sono stati inseriti dei metodi *writer* per scrivere direttamente le proprie informazioni su un file passato come parametro. Queste funzioni sono poi gestite dal metodo di scrittura della classe *IODatabase*.

2.1.7 Classe LinQedInAmministratore

Questa classe è un'interfaccia che fornisce i metodi che un utente di tipo amministratore ha a disposizione per interagire con il database di utenti. Dà quindi

la possibilità di inserire un utente, di rimuoverlo, di cercarlo, sia per id sia per nome e cognome, e di cambiarne la tipologia tra quelle presenti nella gerarchia *Utente*. È la classe che verrà utilizzata dalle classi esterne (esempio le classi della parte grafica) quando vorranno eseguire operazioni possibili da parte di un amministratore.

2.1.8 Classe LinQedInUtente

È la classe avente le stesse funzioni della precedente, solo che per un utente normale, non amministratore. Mette a disposizione i metodi per modificare i propri dati e per inserire o rimuovere le informazioni aggiuntive e i contatti della propria rete. Inoltre include anche la funzione di ricerca di un altro utente, richiamando l'opportuno metodo definito nella classe *Utente*.

2.2 Parte grafica

La parte grafica si occupa dell'interazione dell'utente con la parte logica. Essa mette a disposizione una serie di interfacce grafiche che comunicano con il database e le classi a esso annesse e che permettono all'utente di interagire con tutti gli aspetti del programma.

2.2.1 Classe QtFinestraPrincipale

È la classe principale della parte grafica e costruisce la finestra iniziale del programma. Permette di inserire il proprio id e accedere come utente normale, oppure di accedere come amministratore per poi aprire le rispettive interfacce.

2.2.2 Classe QtFinestraAmministratore

È la classe dedicata all'interfaccia grafica relativa ad un utente amministratore. In questa finestra è possibile visualizzare tutti gli utenti inseriti nel database LinQedIn, inserirne uno nuovo, cercarlo o eliminarlo. È inoltre possibile cambiare la tipologia. Dopo aver selezionato un utente è possibile visualizzarne le sue informazioni tramite un apposito bottone che aprirà una finestra (descritta in seguito) nella quale verranno mostrati tutti i dati dell'utente, senza ovviamente la possibilità di modificarli. Dalla finestra di amministratore c'è anche la possibilità di caricare e salvare il database su file.

2.2.3 Classe QtFinestraUtente

Questa classe rappresenta la finestra dedicata a un Utente che utilizza LinQedIn. Al suo interno un utente può visualizzare i propri dati, le proprie informazioni aggiuntive e i contatti presenti nella sua rete. Ha inoltre la possibilità di modificare i dati personali e salvarli tramite un apposito bottone e di aggiungere o rimuovere le informazioni aggiuntive. Quest'ultimo aspetto porterà all'apertura di una nuova finestra a seconda del tipo di informazione da aggiungere, la quale consentirà l'inserimento dei dati attraverso degli appositi campi. È presente inoltre un apposito bottone per aprire una finestra dedicata a cercare gli utenti nel database LinQedIn e a visualizzare le informazioni dell'utente cercato, informazioni che saranno disponibili diversamente a seconda del tipo di utente che

le visualizza. Una volta trovato l'utente sarà anche possibile aggiungerlo alla propria rete di contatti.

2.2.4 Classe QtFinestraUtenteVisualizzazione

Questa classe è stata accennata in precedenza e permette la creazione dell'interfaccia per visualizzare i dati di un utente. Nella finestra è possibile visualizzare tutte le informazioni personali, le informazioni aggiuntive tramite una visualizzazione ad albero, e la rete di contatti.

2.3 Funzionamento

Il funzionamento dell'applicazione è in parte già stato spiegato precedentemente. All'apertura di presenta una finestra dalla quale possiamo registrare un nuovo utente, accedere utilizzando un id già registrato, oppure accedere come amministratore. Nella finestra di registrazione si devono inserire i dati richiesti nei rispettivi campi dati e poi premere l'apposito bottone per confermare la registrazione. Nella parte dell'utente è possibile visualizzare i propri dati e modificarli. Per salvare le informazioni personali, è necessario premere il pulsante "Salva" dopo le modifiche. Per inserire un'informazione aggiuntiva basta premere il pulsante "+" della rispettiva tabella e inserire i dati nella finestra apposita che si aprirà. Per eliminare un elemento, invece, occorre selezionarlo e premere il pulsante "-". Questo vale anche per la rete di utenti, con l'unica differenza che per aggiungere una persona è necessario prima cercarla nel database tramite l'apposito bottone e successivamente aggiungerla. Nella parte amministratore, si presenta nel mezzo un riquadro in cui sono presenti gli id degli utenti registrati. È possibile fare doppio clic su uno di essi per inserirlo nell'apposito riquadro a sinistra e quindi utilizzarlo per cambiare tipo, cercarlo o eliminarlo dal database. Sempre da questa finestra è possibile premere il bottone apposito per inserire un nuovo utente nello stesso modo in cui si registra un nuovo utente, e visualizzare i dati dell'utente selezionato nel riquadro centrale. In ultima c'è la possibilità di caricare e salvare il database su file esterni.

CAPITOLO 3

NOTE PROGETTUALI

Il progetto è stato realizzato, compilato e testato con QtCreator e compilatore qt di versione 5.3, su una macchina con sistema operativo OSX Yosemite 10.10. È stato inoltre testato nei computer di laboratorio dell'università con la medesima versione di qt.