Maggiolo Giorgio
Matricola: 610338
Anno accademico: 2010/2011

Relazione progetto Legami©

Studente: Giorgio Maggiolo
Matricola: 610338
Anno accademico: 2010/2011

Maggiolo Giorgio
Matricola: 610338
Anno accademico: 2010/2011

Audaces fortuna iuvat
(La fortuna aiuta gli audaci)
Virgilio, Eneide X, 284

Sommario

Introduzione e note di sviluppo

Legami© è un programma per la gestione di contatti di tipo lavorativo, ovvero un "social network" (con *social network* non viene inteso un sito internet, ma viene presa la sua accezione più generale di "*rete sociale*") per permettere le interazioni a livello professionale in modo da poter mettere in comunicazione offerta e richiesta di lavoro, oltre che a mettere in contatto enti differenti fra loro.

Questa relazione vuole essere un documento di presentazione del progetto, senza andare troppo nello specifico tecnico, quanto piuttosto descrivere ed esaminare le scelte implementative che sono state attuate.
Ovviamente, però, alcuni dati tecnici dovranno essere dati e per questo motivo sono stati esposti qui di seguito.

Variabili di sviluppo

- **Sistema operativo:** *Windows 7 Home Edition Service Pack 1 64-bit, Ubuntu 11.04, Ubuntu 10.04*
- **Compilatore:** *MinGw-gcc440-1* e *i486-linux-gnu-4.4.3*
- **Versione libreria Qt©:** 4.7.4(32 bit) e 4.6.2

Note sul progetto ed installazione

La cartella del progetto è divisa in quattro parti:

- La "radice" contiene i file *Legami.pro*, *main.cpp*, e la relazione. Il database, mancante alla consegna, verrà creato automaticamente al primo avvio assieme all'utente amministratore del sistema:
  - Username: root
  - Password: "there is no password" (senza virgolette)
- La cartella "*lib*" contiene i file header e i sorgenti della parte logica del programma, ovvero tutte quelle classi che si occupano della gestione vera e propria delle informazioni e dei dati
- La cartella "*gui*" contiene i file header e i file sorgenti della parte grafica del programma, mera interfaccia per la parte logica
- La cartella "*translation*" contiene i file di traduzione del programma.

Legami© è stato sviluppato nativamente come programma multipiattaforma, per cui non ci sono particolari indicazioni per quanto riguarda la compilazione e l'esecuzione in alcun sistema operativo.
Nel caso dovesse rendersi necessario rigenerare il file di progetto utilizzando il comando "*qmake –project*", si dovrà aggiungere la stringa "*QT += xml*" in quanto, diversamente, non verrebbero incluse le librerie Qt responsabili della gestione dei file xml e quindi il programma risulterebbe non compilante.
Inoltre la cartella "*translation*" non è necessaria ai fini dell'esecuzione, ma consigliata per godere di una migliore esperienza nell'uso di Legami©.

Logica del programma

La "logica del programma" è quella parte di Legami© deputata alla memorizzazione e all'elaborazione dei dati che vengono immessi dall'utente.
Si trova nella cartella *./lib/* della cartella di progetto ed è strutturata in 13 classi.
L'idea che è stata seguita è divergente rispetto a quanto proposto dal Prof. Ranzato, in quanto non segue minimamente le specifiche minime richieste, anche se riprendono alcuni concetti espressi dal professore nelle specifiche e che sono stati implementati.
Quindi ho reimplementato la struttura di base affinché rispecchiasse la mia idea di questo programma che mi sono fatto basandomi sui già esistenti social network Linkedin e Twitter.

Descrizione delle classi:

Legami: è la madre di tutte le classi logiche, se così possiamo definirla. E' responsabile di contenere il database di tutti gli utenti, di tutti i gruppi e di tutti i messaggi, oltre a tenere memoria di quale account sia "loggato" in un determinato momento. Contiene diverse funzionalità come la ricerca divisa in varie tipologie, inserimento o rimozione di messaggi, gruppi e utenti. Tiene in memoria anche il puntatore alla classe deputata alla lettura/scrittura nel database.
Loader: questa classe si occupa della lettura e scrittura del database in un file xml. È stata scelta questa implementazione del database perché più semplice rispetto ad un database vero e proprio realizzato in SQLite, in quanto più ampiamente documentato. (In fondo a questa sezione si troverà una struttura di esempio del file xml)
Account / CompanyAccount / UserAccount: rispetto alle specifiche, ho deciso di creare la gerarchia degli account a partire dalla divisione di questi in account utente e account aziendali. Un *account utente* (UserAccount)è l'account che viene creato da un qualsiasi utente generico, mentre l'*account aziendale* (CompanyAccount) è un account che viene creato, generalmente, dalla sezione "Marketing" di un'azienda che vuole creare un profilo dell'azienda stessa all'interno del portale. La classe Account è una classe virtuale pura che serve come classe base per CompanyAccount e UserAccount.
Gli *amminstratori* di Legami©, invece che essere degli account a parte, sono semplicemente degli UserAccount con un campo dati che indica il fatto che sono amministratori di Legami© o meno.
CompanyInfo / UserInfo: queste due classi contengono i campi informativi dei due account.

- **Companyinfo:**
  - Nome
  - Indirizzo
  - Tipologia dell'azienda: è possibile inserire dei "tag" che identificano un'azienda (ex. Un'azienda informatica inserirà "informatica" nell'elenco delle sue tipologie)
- **UserInfo:**
  - Nome

- Cognome
- Data di nascita
- Luogo di nascita
- Numero di telefono
- Indirizzo Email

Username: come dice il nome stesso, questa classe contiene le credenziali di accesso a Legami©.

- **Username**:
  - Username
  - Password

Experience: questa classe rappresenta un'esperienza che può essere di tipo formativa o lavorativa. Può appartenere solo ad un utente (non avrebbe senso che un'azienda avesse "esperienze").

- **Experience:**
  - Tipo (Formativa o Lavorativa)
  - Nome (Ex. Laurea in xxx)
  - Data dell'esperienza
  - Descrizione

Group: un gruppo è una *collezione* di utenti sotto un medesimo nome e sotto una medesima descrizione. All'atto di creazione di un nuovo gruppo, l'account che l'ha creato ne diventa l'amminstratore, però è possibile che esistano gruppi senza alcun amminstratore. In questo caso solo un amministratore di Legami© potrà effettuarvi modifiche, tramite il pannello di amminstrazione.

- **Group**
  - Nome
  - Descrizione
  - Elenco degli utenti
  - Elenco degli amministratori

Message: rappresenta un singolo messaggio che viene inviato da un mittente verso un destinatario, con un determinato oggetto ed un determinato testo. Inoltre questa classe tiene traccia del fatto che il destinatario abbia letto o meno il messaggio.

- **Message:**
  - Mittente
  - Destinatario
  - Oggetto
  - Testo
  - Letto

Payment: rappresenta una "richiesta di upgrade". Infatti il sistema non permette l'upgrade automatico di un account, ma richiede che sia un amministratore ad autorizzare l'upgrade.

- **Payment:**
  - Account richiedente
  - Tipologia richiesta
  - Data della richiesta
  - Approvato

Struttura tipo file xml

```
<!DOCTYPE legami>
<legami>
 <users>
  <user> (UserAcccount)
   <userdata>
    <username></username>
    <password></password>
    <type></type>
    <acctype></acctype>
    <admin></admin>
   </userdata>
   <infos>
    <name></name>
    <surname></surname>
    <daybirth></daybirth>
    <monthbirth></monthbirth>
    <yearbirth></yearbirth>
    <birthplace></birthplace>
    <telnum></telnum>
    <email></email>
   </infos>
   <experiences>
    <experience>
     <exptype></exptype>
     <expname></expname>
     <dayexp></dayexp>
     <monthexp></monthexp>
     <yearexp></yearexp>
     <expdescr></expdescr>
    </experience>
   </experiences>
  </user>
  <user> (CompanyAccount)
   <userdata>
    <username></username>
    <password></password>
    <type></type>
    <acctype></acctype>
   </userdata>
   <companydata>
    <name></name>
    <address></address>
    <cotype></cotype>
   </companydata>
  </user>
</users>
 <connections>
  <connection>
   <applicant>root</applicant>
   <applied>dad</applied>
  </connection>
  <connection>
```

```
    <applicant></applicant>
    <applied></applied>
   </connection>
  </connections>
  <groups>
   <group>
    <name></name>
    <descr></descr>
    <members>
     <member></member>
    </members>
    <admins>
     <admin></admin>
    </admins>
   </group>
  </groups>
  <messages>
   <message>
    <sender></sender>
    <recever></recever>
    <object></object>
    <text></text>
    <read></read>
   </message>
  </messages>
  <payments>
   <payment>
    <applicant></applicant>
    <typerequested></typerequested>
    <dayrequest></dayrequest>
    <monthrequest></monthrequest>
    <yearrequest></yearrequest>
    <approved></approved>
   </payment>
  </payments>
</legami>
```

Grafica del programma

La grafica di Legami© è stata sviluppata utilizzando le librerie Qt 4.6.2.
Ho scelto di sviluppare il codice in modo tale da renderlo modulare.
Infatti tutti i widget che vengono mostrati all'utente sono in realtà composti (tranne quelli più ovvi) da 1 o più widget, che vengono creati e uniti nel widget che poi verrà mostrato all'utente.
Questo per incentivare il riutilizzo del codice, in modo che, nel caso si vogliano sviluppare ulteriori funzionalità, si possano utilizzare i widget creati e combinarli fra di loro.

Poiché la grafica comprende 40 widget, non riesco ad includere una lista dettagliata di ogni singolo file con relativa descrizione. Per cui ho deciso di inserire una lista con solo i widget "principali", ovvero i contenitori di widget.

Descrizione dei widget "principali"

Addnewcontact: deriva da *QDialog*. Aggiunge un nuovo account alla lista dei contatti dell'account loggato.
Addrequestpayment: deriva da *QDialog*. Mostra le caratteristiche delle tipologie di account (basic, business, executive) e la possibilità di emettere una richiesta di upgrade.
Inboxmessges: deriva da *QWidget*. Mostra i messaggi in arrivo per l'account loggato permettendo di leggerne il contenuto.
Legamilogin: deriva da *QDialog*. Permette il login di un account, chiamando la funzione login() di legami.
Legamimainwindow: deriva da *QMainWidget*. È la finestra principale. Crea un'istanza di legami e carica il database, oltre a caricare i file di traduzione.
Modifycompanyprofile: deriva da *QWidget*. Permette la modifica di un account aziendale
Modifyuserprofile: deriva da *QWidget*. Permette la modifica di un account utente.
Newmessage: deriva da *QDialog*. Permette l'invio di un nuovo messaggio, con le limitazioni derivate dalla tipologia di account.
Outboxmessages: deriva da *QWidget*. Mostra i messaggi inviati per l'account loggato permettendo di leggerne il contenuto.
Reguser: deriva da *QDialog*. Permette l'iscrizione di un nuovo account.
Showcompanyprofile: deriva da *QWidget*. Mostra il profilo di un account aziendale.
Showpayments: deriva da *QWidget*. Mostra le richieste di upgrade effettuate dall'account loggato.
Showuserprofile: deriva da *QWidget*. Mostra il profilo di un account utente.
Viewotheruser: deriva da *QWidget*. Permette la visualizzazione di un account diverso da quello attualmente loggato.