Programmation d'interface

Projet : Contact Book

Projet n° 2

DIBASSI Brahima 19005521

KANOUTE Daouda 19000407

Dimanche 20 Décembre 2020

Contents

1 Introduction

1.1 Présentation

Notre depot git : https://github.com/Bibi210/Contact_Book
Projet, réalisé dans le cadre du cours de programmation d'interface de L2 avec Mr.Halim Djerroud (hdd@ai.univ-paris8.fr).

Pour le projet de fin de semestre de ce cours nous avons décidé de prendre le projet 2, il n'y a pas de raison particulière a cela nous avons juste préferé celui ci au jeu de dames.

1.2 Énoncé du projet

On souhaite écrire une application de gestion de contacts téléphoniques avec une interface graphique. Pour chaque personne on sauvegarde son nom et prénom . . ..
Chaque contact peut avoir plusieurs numéros de téléphones. Les informations à stocker pour chaque contacte sont
les suivantes :
—
Nom : Doe
Prénom : Jane
Adresse : xx
CP : xxxxx
Type contact : Pro, Perso, Autre
e-mail : xx@xx.xxx
Num_1 : 0102030405
Num_2 : 0102030406
Num_n : 0102030409

2 Fonctionnment

- **initList()**:Permet d'initialiser notre **GTK TREE VIEW**,
  ajoute des colonnes et si il y'a des contacts dans notre base de donnée,
  il les ajoute a la vue ainsi que dans notre hashtable.
  Prend en argument une liststore, un widget qui fera office
  de gtk$_{t}ree_{v}iewainsiqu'unbuilderquipermettrad'ajouter$
  $latreeviewdansnotrescrollwindowcreeravecglade.$

- **addToList()**:Ajoute des element a notre treeview grace a la
  structure de **gtk entry** passé en argument tout en recuperant la value avant,
  ajoute aussi dans la hashtable tout en discriminant les contacts
  n'ayant pas de nom,prenom ou numero de telephone, si tel est le cas
  affiche une boite de dialog annoncant l'erreur.
  Prend un argument un gpointeur qui sera ensuite changé en structure.

- **ShowModal()**:Affiche une boite de dialog utile pour ajouter un contact,
  au clic du boutton ajouter, la fonction addToList est appellée avec comme
  argument
  une structure de gtk entry. Ne prend pas d'argument.

- **detailsView()**:Permet d'afficher les details du contact dans la vue de droite.
  Utilise la selection de la treeview pour rechercher dans la hashtable un
  contact,
  ceux grace aux nom+prenom+numero1 qui servent de clé a la hashtable.
  A la suite de quoi il changera les labels avec les value correspondante.
  prend un gpointeur de contact, qui sont en realité des labels.

- **removeItem()**:Grace a la selection de la treeview, permet de supprimer un
  element
  de celle-ci ainsi que de la hashtable.
  prend en argument la selection de treeview en gpointeur.

- **EditMode(n)**:Permet d'editer un contact grace a la selection.
  cet fonction appelle une boite de dialog au clic du boutton edit
  celle ci est prerempli avec les donnée du contact sélectionner
  lorsque l'edition est validé, elle appelle la fonction removeContact puis addToList
  l'original est donc supprimé pour laisser place a la modification.
  Ne prend pas d'argument.

- **Search()**: Prend la hashtable puis regarde pour chaque contact si la value
  de

l'entrée correspond a l'un des value de la hashtable.
Prend en argument la searchBar.

- **updateLabel()**: Met ajour le nombre de contact dans la hashtable
  Ne prend pas d'argument.

- **contactBookQuit()**: Quitte le programme et met les element de la hashtable dans la base de donnée.
  Ne prend pas d'argument.

3 Fonctionnalite

3.1 Fonctionalité implémenter

- Ajouter des contacts, avec obligatoirement Nom, Prenom et Numero et combobox.
- Supprimer des contacts.
- Modifier un contact.
- Le clique sur un contact provoque l'affichage des informations du contact.
- Afficher le nombre de contact
- Recherche de contact par nom prenom ou numeros
- Sauvegarder les contacts.

3.2 Utile a savoir

- La base de donnée n'est utilisé que deux fois, une fois au demarrage de l'application pour restituer les contacts dans la hashtable
  et une autre a sa fermeture pour recuperer les elements de la hashtable.

- La modifications se fait par suppression et ajout d'un nouveau contact avec les nouvelles value.

4 Fonction importante

```
static void initList(GtkWidget *listViewe, GtkListStore *listStore,
GtkBuilder *builder)
{
    GtkCellRenderer *cellRenderer;
    GtkTreeViewColumn *column;
    GtkWidget *scrollView;
    GtkTreeIter iter;
    gchar column_names[3][10] = {"prenom", "nom", "numero"};
    GList *contact_list = NULL;
    GList *UnType = NULL;
    t_contact_hash *un_contact = NULL;

    scrollView = GTK_WIDGET(gtk_builder_get_object(builder, "contact_list"));
    cellRenderer = gtk_cell_renderer_text_new();
    for (guint i = 0; i < 3; i++)
    {
        column = gtk_tree_view_column_new_with_attributes(column_names[i],
                                                          cellRenderer,
                                                          "text", i,
                                                          NULL);
        gtk_tree_view_append_column(GTK_TREE_VIEW(listView), column);
    }

    // Gestion auto du scroll
    gtk_scrolled_window_set_policy(GTK_SCROLLED_WINDOW(scrollView),
                                   GTK_POLICY_AUTOMATIC,
                                   GTK_POLICY_AUTOMATIC);
    gtk_container_add(GTK_CONTAINER(scrollView), listView);
    listStore = gtk_list_store_new(N_COLUMN, G_TYPE_STRING, G_TYPE_STRING,
     G_TYPE_STRING);
    gtk_tree_view_set_model(GTK_TREE_VIEW(listView),
                            GTK_TREE_MODEL(listStore));

    for (GList *encours = data_base_retrive(); encours != NULL;
     encours = encours->next)
    {
        UnType = encours->data;
        contact_list = g_list_append(contact_list, cast_glist_to_contact(UnType));
    }

    for (GList *contact = contact_list; contact != NULL;
     contact = contact->next)
```

```
    {
        un_contact = contact->data;
        gchar *key = g_strdup(g_strconcat(un_contact->Nom, un_contact->Prenom,
         un_contact->number1, NULL));
        g_hash_table_insert(hashContact, key, un_contact);
        gtk_list_store_append(listStore, &iter);
        gtk_list_store_set(listStore, &iter,
                           NAME_COLUMN, un_contact->Prenom,
                           LAST_NAME_COLUMN, un_contact->Nom,
                           NUMBER_COLUMN, un_contact->number1,
                           -1);
        nb_contact = g_hash_table_size(hashContact);
    }
    updateLabel(GTK_LABEL(number_of_contact));
    data_base_clear();
}
```

5 Difficultés, solutions et organisation

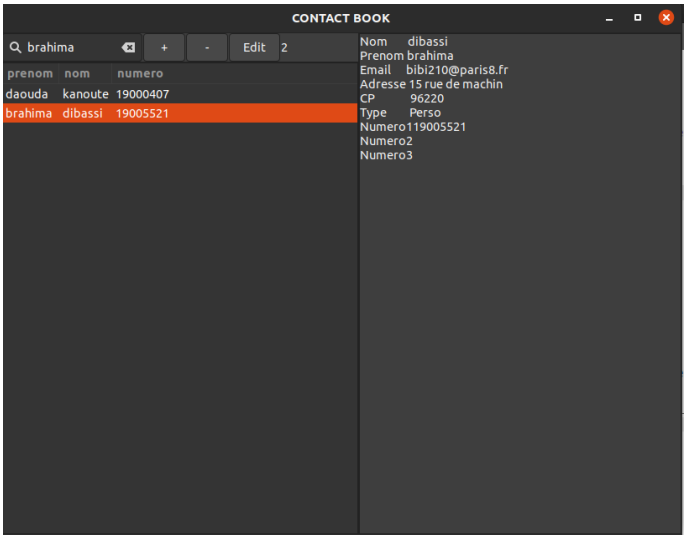
5.1 Difficulté principal

Lors de l'implémentation de la bar de recherche notre principale difficulté fut la recherche par critère multiple en effet certes GTKTREEVIEW possède une recherche intégrer mais dans notre cas cela nous permettait pas la recherche par numéro et par nom prénom en simultané pour résoudre ce problème nous nous somme orienter vers l'idée de déplacer la sélection vers la recherche ce qui nous a grandement simplifier le travaille

5.2 Organisation

Notre organisation a été assez simple. Au début du projet, nous nous sommes attribuée des taches de facons a ce que nous puissions tout les deux avoir de quoi faire sur le projet et que personne ne soit laissé pour compte. Nous avons ensuite creer un projet git pour que nous puissions travailler ensemble facilement. Nous travaillions en ensemble en conference sur Jitsi afin de pouvoir s'entraider, a la fin de chaque conference, nous nous faisions un rapport sur ce qu'il reste a faire pour chacun. Une fois que nous avons eu notre premieres versions du projet, nous nous sommes reuni afin de rendre notre code plus propres pour le merge final.

6 Visuel


CONTACT BOOK
brahima
+
-
Edit
2
prenom nom numero
daouda kanoute 19000407
brahima dibassi 19005521
Nom dibassi
Prenom brahima
Email bibi210@paris8.fr
Adresse 15 rue de machin
CP 96220
Type Perso
Numero1 19005521
Numero2
Numero3