EXAMEN D'INITIATION AUX INTERFACES GRAPHIQUES
Institut Galilée - Université Paris 13 **Mardi 21 juin 2011 – 14h-16h**
Licence 1 **Aucun document autorisé**

Présentation

Vous disposez d'une bibliothèque *"calculatrice"* représentant une calculatrice pour nombres rationnels (*cf.*, l'extrait du fichier d'entête qui est joint au sujet). Pour cette calculatrice, on dipose de deux nombres rationnels, appelés *"résultat"* et *"opérande"*, ainsi que d'une opération, appelée *"opérateur"*. Initialement, les nombres rationnels de la calculatrice valent $0/1$, l'opération est l'addition. L'utilisateur doit pouvoir initialiser la calculatrice (aux valeurs sus-mentionnées), ou l'évaluer. L'évaluation consiste à calculer le résulat de l'opération rationnelle *"résultat opérateur opérande"*, puis à placer ce résultat dans *"résultat"*.

Partie A : prise en mains de la bibliothèque *"calculatrice"*

1. Suite à une mauvaise manipulation, il faut réécrire les fonctions suivantes de la bibliothèque *"calculatrice"* :
   (a) `calculatrice_get_res_num`, `calculatrice_set_opa_den`, `calculatrice_get_ope` ;
   (b) `calculatrice_affiche`, `calculatrice_initialise`.
2. Donner (en justifiant) la valeur de la variable `c→res` pour chaque ligne de la succession d'instructions :

```
1  s_calculatrice c;
2  calculatrice_initialiser(& c);
3  calculatrice_set_opa_num(& c, 3);
4  calculatrice_evalue(& c);
5  calculatrice_set_opa_den(& c, 2);
6  calculatrice_evalue(& c);
```

Partie B : interface graphique

Vous devez développer l'interface graphique de la calculatrice, dont une ébauche est fournie dans la figure 1. Lorsque l'application est lancée, la calculatrice (du modèle comme sa représentation via l'interface) doit être initialisée. L'utilisateur doit ensuite pouvoir, par l'intermédiaire de l'interface, choisir un nouvel opérateur (en cliquant sur l'un des quatre opérateurs $+$, $-$, $*$, $/$), évaluer l'expression courante (en cliquant sur "Eval"), ré-initialiser la calculatrice (en cliquant sur "Init"), modifier l'opérande. L'interface est à réaliser en gtk. Vous disposez des "widgets" gtk suivants : `GtkWindow` (fenêtre), `GtkHBox` (boîte horizontale), `GtkVbox` (boîte verticale), `GtkButton` (bouton), `GtkLabel` (libellé), `GtkEntry` (zone de saisie).

opa :	0	|	1
res :	0	|	1
ope :	+		
+	−	*	/
Eval	Init		

Figure 1: Ébauche de l'interface attendue

1. Composants graphiques gtk. Parmi les "widget" gtk listés ci-dessus, lesquels :
   (a) sont interactifs (le cas échéant, préciser le type d'interaction) ?
   (b) sont des conteneurs (*ie.*, peuvent contenir au moins un autre composant graphique) ?
   (c) sont des conteneurs pouvant contenir au moins deux composants graphiques) ?
2. Fonctions gtk. Que font les fonctions gtk suivantes :
   (a) `gtk_container_add` ?
   (b) `gtk_box_pack_start` ?
   (c) `gtk_signal_connect` ?
   (d) `gtk_main_quit` ?

3. Conception de l'interface en gtk. En utilisant exclusivement les "widget" précédemment listés :
   (a) quels sont les objets graphiques impliqués, et en quel nombre (*ex.*, 3 boutons, 2 libellés, etc.) ?
   (b) comment ces objets sont-ils imbriqués entre-eux (*ex.*, le bouton `toto` dans la fenêtre `tata`, etc.) ?
   (c) pour chaque imbrication d'un objet dans un autre, indiquer la fonction gtk à utiliser (en justifiant).
4. Comportement de l'interface gtk :
   (a) Indiquer pour chaque composant interactif son état d'activité en fonction de l'état de la calculatrice.
   (b) Indiquer pour chaque composant `GtkLabel` utilisé, son évolution au cours de l'utilisation de la calculatrice : par quelle action de l'utilisateur sa valeur peut-elle être modifiée, quelle nouvelle valeur sera prise.
5. Sollicitations faites au modèle. Selon vous :
   (a) Quelle(s) fonction(s) de la bibliothèque "calculatrice" faut-il appeler préalablement au lancement de l'interface graphique ?
   (b) Quelle(s) fonction(s) de la bibliothèque "calculatrice" faut-il appeler à l'issue d'un clic opéré sur "Eval" ? Sur "Init" ?
   (c) Quelle(s) fonction(s) de la bibliothèque "calculatrice" faut-il appeler à l'issue d'une modification par l'utilisateur, via l'interface, du numérateur de l'opérande ?
6. Pour faire en sorte que votre interface ait le comportement décrit dans les réponses précédentes, indiquer :
   (a) le proptotype et la vocation des fonctions que vous envisagez devoir écrire (mais sans écrire les fonctions) ;
   (b) l'ensemble des appels que vous envisagez de faire à la fonction `gtk_signal_connect` ;
   (c) les éventuelles structures de données que vous envisagez de définir pour faire appel à ces fonctions.

Partie C : quelques révisions en langage C et gtk

1. Le programme suivant produit une erreur à la compilation ; quelles sont les instructions fautives ?

```
1   #include <gtk/gtk.h>
2   int main (int argc, char *argv[ ])
3   {
4       GtkWindow* pFenetre = NULL;
5       gtk_init(NULL, NULL);
6       pFenetre = gtk_window_new(GTK_WINDOW_TOPLEVEL);
7       gtk_window_set_title(pFenetre, "toto");
8       gtk_widget_show_all(GTK_WIDGET(pFenetre));
9       gtk_main();
10      return 0;
11  }
```

2. Le programme suivant compile, mais produit des erreurs à l'exécution. Lesquelles, et pourquoi ?

```
1   #include <gtk/gtk.h>
2   int main (int argc, char *argv[ ])
3   {
4       GtkWidget* pFenetre = NULL;
5       GtkWidget* pBoutonQuitter = NULL;
6       gtk_init(NULL, NULL);
7       pFenetre = gtk_window_new(GTK_WINDOW_TOPLEVEL);
8       pBoutonQuitter = gtk_label_new("_Quitter");
9       gtk_container_add(GTK_CONTAINER(pFenetre), pBoutonQuitter);
10      gtk_window_set_title(GTK_WINDOW(pFenetre), "tata (pour changer)");
11      gtk_button_set_label(GTK_BUTTON(pBoutonQuitter), "titi (soyons fous !)");
12      gtk_widget_show_all(GTK_WIDGET(pFenetre));
13      gtk_main();
14      return 0;
15  }
```