```
#include <stdlib.h>

#include "queen.h"


void callback_destroy(GtkWidget * widget, gpointer data)
{
    gtk_main_quit();
}

void callback_modif(GtkWidget * widget, gpointer data)
{
    queen_t* tmp ;
    tmp = data ;
    modif_win_show(tmp->interface->win_modif);
}

////////////////////////////////////AUTEUR CALLBACK//////////////////////////
/////////////////
void callback_auteur(GtkWidget * widget, gpointer data)
{
    queen_t* tmp ;
    tmp = data ;
    // Remplissage de la fenÃªtre avec le contenu de auteur
    auteur_win_fill(tmp->interface->win_auteur, tmp->projet->video->auteur ) ;

    //Apparition de la fenÃªtre
    auteur_win_show(tmp->interface->win_auteur);

}

void callback_auteur_modify_name(GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_auteur->button_modif_1 == 0 )
        auteur_button_modify_name(tmp->interface->win_auteur, tmp->projet->video
->auteur ) ;
        else
        auteur_button_modify_ok_name(tmp->interface->win_auteur, tmp->projet->vi
deo->auteur ) ;

}

void callback_auteur_modify_first_name(GtkWidget* widget, gpointer data) {

        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_auteur->button_modif_2 == 0 )
        auteur_button_modify_first_name(tmp->interface->win_auteur, tmp->projet-
>video->auteur ) ;
        else
        auteur_button_modify_ok_first_name(tmp->interface->win_auteur, tmp->proj
et->video->auteur ) ;
}


void callback_auteur_modify_email (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;
```



```
        if ( tmp->interface->win_auteur->button_modif_3 == 0 )
        auteur_button_modify_email(tmp->interface->win_auteur, tmp->projet->vide
o->auteur ) ;
        else
        auteur_button_modify_ok_email(tmp->interface->win_auteur, tmp->projet->v
ideo->auteur ) ;
    }
////////////////////////////////////VIDEO CALLBACK //////////////////////////////
/////////

void callback_video(GtkWidget * widget, gpointer data)
{
    queen_t* tmp ;
    tmp = data ;
    // Remplissage de la fenÃªtre avec le contenu de auteur
    video_win_fill (tmp->interface->win_video, tmp->projet->video ) ;
    //Apparition de la fenÃªtre
    video_win_show (tmp->interface->win_video) ;
}

void callback_video_modify_name_ruche (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_video->cnt_modif_name_ruche == 0 )
        video_button_modify_name_ruche_modif(tmp->interface->win_video, tmp->pro
jet->video ) ;
        else
        video_button_modify_name_ruche_ok(tmp->interface->win_video, tmp->projet
->video ) ;
    }

void callback_video_modify_n_ruche (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_video->cnt_modif_n_ruche == 0 )
        video_button_modify_n_ruche_modif(tmp->interface->win_video, tmp->projet
->video ) ;
        else
        video_button_modify_n_ruche_ok(tmp->interface->win_video, tmp->projet->v
ideo ) ;
    }


void callback_video_modify_n_cadre (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_video->cnt_modif_n_cadre == 0 )
        video_button_modify_n_cadre_modif(tmp->interface->win_video, tmp->projet
->video ) ;
        else
        video_button_modify_n_cadre_ok(tmp->interface->win_video, tmp->projet->v
ideo ) ;
    }

void callback_video_modify_description (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_video->cnt_modif_description == 0 )
```

```
        video_button_modify_description_modif(tmp->interface->win_video, tmp->pr
ojet->video ) ;
        else
        video_button_modify_description_ok(tmp->interface->win_video, tmp->proje
t->video ) ;
    }

////////////////////////////// CAMERA CALLBACK ////////////////////////////////
/////

void callback_camera(GtkWidget * widget, gpointer data)
{
    queen_t* tmp ;
    tmp = data ;
    // Remplissage de la fenÃªtre avec le contenu de camÃ©ra
    camera_win_fill(tmp->interface->win_camera, tmp->projet->camera ) ;

    //Apparition de la fenÃªtre
    camera_win_show(tmp->interface->win_camera);
}

void callback_camera_modify_name (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_camera->cnt_modif_name == 0 )
        camera_button_modify_name_modif(tmp->interface->win_camera, tmp->projet-
>camera ) ;
        else
        camera_button_modify_name_ok(tmp->interface->win_camera, tmp->projet->ca
mera ) ;
    }

void callback_camera_modify_model (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_camera->cnt_modif_model == 0 )
        camera_button_modify_model_modif(tmp->interface->win_camera, tmp->projet
->camera ) ;
        else
        camera_button_modify_model_ok(tmp->interface->win_camera, tmp->projet->c
amera ) ;
    }

void callback_camera_modify_serial (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_camera->cnt_modif_serial == 0 )
        camera_button_modify_serial_modif(tmp->interface->win_camera, tmp->proje
t->camera ) ;
        else
        camera_button_modify_serial_ok(tmp->interface->win_camera, tmp->projet->
camera ) ;
    }

void callback_camera_modify_type (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_camera->cnt_modif_type == 0 )
```



```
        camera_button_modify_type_modif(tmp->interface->win_camera, tmp->projet-
>camera ) ;
        else
        camera_button_modify_type_ok(tmp->interface->win_camera, tmp->projet->ca
mera ) ;
    }

void callback_camera_modify_description (GtkWidget* widget, gpointer data) {
        queen_t* tmp ;
        tmp = data ;

        if ( tmp->interface->win_camera->cnt_modif_description == 0 )
        camera_button_modify_description_modif(tmp->interface->win_camera, tmp->
projet->camera ) ;
        else
        camera_button_modify_description_ok(tmp->interface->win_camera, tmp->pro
jet->camera ) ;
    }


////////////////////////////////////////////////// TAG CALLBACK ////////////////
//////////////////////////////////
void callback_win_tag(GtkWidget * widget, gpointer data)
{
    queen_t* tmp ;
    tmp = data ;

    tag_win_show(tmp->interface->win_tag);
}

void callback_quit_tag(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    gtk_window_close (GTK_WINDOW(tmp->interface->win_tag->window));
}
////////////////////////////////////////////////// INFO CALLBACK /////////////
//////////////////////////////
void callback_info(GtkWidget *widget, gpointer data)
{
    queen_t* tmp ;
    tmp = data ;
    info_win_show(tmp->interface->win_info) ;
}

////////////////////////////////////////////////// COLOR CALLBACK /////////////
////////
void callback_color(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    color_win_show ( tmp->interface->win_color ) ;
}

////////////////////////////////////////////////// WIN MODIF CALLBACK ////////
/////////////
void callback_win_cross(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    cross_win_show ( tmp->interface->win_cross ) ;
}

void callback_win_box(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
```

```
    tmp = data ;
    box_win_show ( tmp->interface->win_box ) ;
}

void callback_quit_modif(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    gtk_window_close (GTK_WINDOW(tmp->interface->win_modif->window));
}
////////////////////////////////////////////////////// WIN FILE //////////////
//////////

void callback_win_file(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    file_win_show ( tmp->interface->win_file ) ;
}
////////////////// WIN CUT //////

void callback_win_cut(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    cut_win_show ( tmp->interface->win_cut ) ;
}


void callback_quit_cut(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    gtk_window_close (GTK_WINDOW(tmp->interface->win_cut->window));
}

/// CROSS
void callback_quit_cross(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    gtk_window_close (GTK_WINDOW(tmp->interface->win_cross->window));
}

///BOX
void callback_quit_box(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    gtk_window_close (GTK_WINDOW(tmp->interface->win_box->window));
}
///CREATE_TAG
void callback_win_create_tag(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    create_tag_win_show ( tmp->interface->win_create_tag ) ;
}

void callback_quit_create_tag(GtkWidget* widget, gpointer data) {
    queen_t* tmp ;
    tmp = data ;
    gtk_window_close (GTK_WINDOW(tmp->interface->win_create_tag->window));
}
////////////////////////////////////////////////////////////********** MAIN**********
******* ////////////////////////////////////////////////////////////
int main(int argc, char *argv[])
{
```



```
//CrÃ©ation de la grande structure
    queen_t * queen;

//Initialisation de GTK
    gtk_init(&argc, &argv);

//DÃ©finition de la grande structure
    queen = queen_new();

//Affichage de la fenÃªtre principale
    main_win_show(queen->interface->win_main);

////////////////////////////////GSNIGAL//////////////////////////////
//FenÃªtre MAIN

    g_signal_connect(queen->interface->win_main->btn_enregistrer,
                            "clicked",
                                G_CALLBACK(callback_modif),
                                queen);                         //Affiche win modif quan
d appui sur bouton


    g_signal_connect(queen->interface->win_main->btn_info,
                            "clicked",
                                G_CALLBACK(callback_info),
                                queen);

    g_signal_connect(queen->interface->win_main->btn_file,
                            "clicked",
                                G_CALLBACK(callback_win_file),
                                queen);


//FenÃªtre AUTEUR
    g_signal_connect(queen->interface->win_auteur->button_modify1,
            "clicked",
            G_CALLBACK(callback_auteur_modify_name),
            queen);


    g_signal_connect(queen->interface->win_auteur->button_modify2,
            "clicked",
            G_CALLBACK(callback_auteur_modify_first_name),
            queen);

    g_signal_connect(queen->interface->win_auteur->button_modify3,
            "clicked",
            G_CALLBACK(callback_auteur_modify_email),
            queen);

//FenÃªtre CAMERA
     g_signal_connect(queen->interface->win_camera->btn_modify_name,
            "clicked",
            G_CALLBACK(callback_camera_modify_name),
            queen);

     g_signal_connect(queen->interface->win_camera->btn_modify_model,
            "clicked",
            G_CALLBACK(callback_camera_modify_model),
            queen);
```

```
    g_signal_connect(queen->interface->win_camera->btn_modify_serial,
            "clicked",
            G_CALLBACK(callback_camera_modify_serial),
            queen);

    g_signal_connect(queen->interface->win_camera->btn_modify_type,
            "clicked",
            G_CALLBACK(callback_camera_modify_type),
            queen);

    g_signal_connect(queen->interface->win_camera->btn_modify_description,
            "clicked",
            G_CALLBACK(callback_camera_modify_description),
            queen);

//FenÃªtre VIDEO
    g_signal_connect(queen->interface->win_video->btn_modify_name_ruche,
            "clicked",
            G_CALLBACK(callback_video_modify_name_ruche),
            queen);

    g_signal_connect(queen->interface->win_video->btn_modify_n_ruche,
            "clicked",
            G_CALLBACK(callback_video_modify_n_ruche),
            queen);

    g_signal_connect(queen->interface->win_video->btn_modify_n_cadre,
            "clicked",
            G_CALLBACK(callback_video_modify_n_cadre),
            queen);

    g_signal_connect(queen->interface->win_video->btn_modify_description,
            "clicked",
            G_CALLBACK(callback_video_modify_description),
            queen);



//FenÃªtre MODIF_WIN
    g_signal_connect(queen->interface->win_modif->btn_an_cross,
            "clicked",
            G_CALLBACK(callback_win_cross),
            queen);

    g_signal_connect(queen->interface->win_modif->btn_an_box,
            "clicked",
            G_CALLBACK(callback_win_box),
            queen);

   g_signal_connect(queen->interface->win_modif->btn_add_tag,
                    "clicked",
                        G_CALLBACK(callback_win_tag),
                        queen);

   g_signal_connect(queen->interface->win_modif->btn_cut,
                    "clicked",
                        G_CALLBACK(callback_win_cut),
                        queen);


   g_signal_connect(queen->interface->win_modif->btn_ann,
                    "clicked",
```



```
                        G_CALLBACK(callback_quit_modif),
                        queen);
//Win_cut
    g_signal_connect(queen->interface->win_cut->btn_ann,
                    "clicked",
                        G_CALLBACK(callback_quit_cut),
                        queen);

//Win_cross
    g_signal_connect(queen->interface->win_cross->btn_ann,
                    "clicked",
                        G_CALLBACK(callback_quit_cross),
                        queen);

    g_signal_connect(queen->interface->win_cross->btn_add_tag,
                    "clicked",
                        G_CALLBACK(callback_win_tag),
                        queen);
//win_box
    g_signal_connect(queen->interface->win_box->btn_ann,
                    "clicked",
                        G_CALLBACK(callback_quit_box),
                        queen);

    g_signal_connect(queen->interface->win_box->btn_add_tag,
                    "clicked",
                        G_CALLBACK(callback_win_tag),
                        queen);
//win_tag
    g_signal_connect(queen->interface->win_tag->btn_create_tag,
                    "clicked",
                        G_CALLBACK(callback_win_create_tag),
                        queen);

    g_signal_connect(queen->interface->win_tag->btn_ann,
                    "clicked",
                        G_CALLBACK(callback_quit_tag),
                        queen);
//win_create_tag
    g_signal_connect(queen->interface->win_create_tag->btn_ann,
                    "clicked",
                        G_CALLBACK(callback_quit_create_tag),
                        queen);



//Fonction attend event.
    gtk_main();

//Suppr des Ã©lÃ©ments
    queen_del(queen) ;

    return 0;

}
```