Programmazione di GUI con GTK – parte 3

by Özcan Güngör
<ozcangungor(at)netscape.net>

About the author:
Attualmente sto facendo il servizio militare come amministratore di sistema e programmatore web con Linux e Oracle.

Abstract:

In questa serie di articoli impareremo come scrivere interfacce grafiche (GUI) usando GTK. Non ho idea di quanto durerà. Per capire questi articoli dovete conoscere le seguenti cose riguardo la programmazione in C:

- Variabili
- Funzioni
- Puntatori

Si raccomanda di leggere gli articoli precedenti:
Programmazione GUI conGTK,
Programmazione GUI con GTK – parte 2 .
Questo articolo è un po' più corto degli altri perché sto facendo il servizio militare.

Toggle Button

Questo bottone assomiglia a un normale bottone ma ha due stati: premuto o non premuto. Per creare un toggle button si deve usare una delle seguenti funzioni:

```
GtkWidget *toggle=gtk_toggle_button_new(void);
GtkWidget *toggle=gtk_toggle_button_new_with_label(const gchar *label);
```

La prima crea un toggle button senza un'etichetta (label), mentre la seconda aggiunge una stringa come etichetta al bottone.
Potete impostare lo stato con la seguente funzione:

```
gtk_toggle_button_set_active (GtkToggleButton *toggle_button,
                              gboolean is_active);
```

dove *toggle_button* è il bottone di cui volete cambiare lo stato e *is_active* è lo stato (0 or 1). Quando è 0 il pulsante è "non premuto"; quando è 1 il pulsante è "premuto".

Per conoscere lo stato del pulsante si può usare questa funzione:

```
gboolean gtk_toggle_button_get_active(GtkToggleButton *button);
```

L'evento "toggled" può essere collegato a un bottone toggle.

Ecco un esempio:

```
#include <gtk/gtk.h>
void togg(GtkWidget *widget, gpointer *data){
   if (gtk_toggle_button_get_active(GTK_TOGGLE_BUTTON(data)))
    g_print("Lo stato è 1\n");
   else
    g_print("Lo stato è 0\n");
}

int main( int argc,char *argv[] )
   {

   GtkWidget *window;
   GtkWidget *button;

   gtk_init (&argc, &argv);

   /* Create a new window */
   window = gtk_window_new (GTK_WINDOW_TOPLEVEL);
   gtk_window_set_title (GTK_WINDOW (window), "Toggle Button");

   /* Connect destroy event to the window. */
   gtk_signal_connect (GTK_OBJECT (window), "destroy",
                       GTK_SIGNAL_FUNC(gtk_main_quit), NULL);

   /* Creates a toggle button */
   button=gtk_toggle_button_new_with_label("Sono un toggle button");

   /* Add the button to window */
   gtk_container_add(GTK_CONTAINER(window),button);

   /* Connect "toggled" event to the button */
   gtk_signal_connect (GTK_OBJECT (button), "toggled",
                       GTK_SIGNAL_FUNC(togg),(gpointer *)button);

   gtk_widget_show(button);
   gtk_widget_show (window);

   gtk_main ();
   return(0);
}
```

Check Button

Il check button (conosciuto anche come check box) è una sotto−classe del toggle button. Può essere usato per scegliere diverse opzioni.
Per creare un check button si usano le seguenti funzioni:

```
GtkWidget* gtk_check_button_new (void);
GtkWidget* gtk_check_button_new_with_label (const gchar *label);
```

Funzionano allo stesso modo delle rispettive funzioni per i toggle button.

Esempio :

```
#include <gtk/gtk.h>
void togg(GtkWidget *widget, gpointer *data){
   if (gtk_toggle_button_get_active(GTK_TOGGLE_BUTTON(data)))
       g_print("Lo stato è 1\n");
   else
    g_print("Lo stato è 0\n");
}

int main( int argc,char *argv[] )
{

    GtkWidget *window;
    GtkWidget *button;

    gtk_init (&argc, &argv);

    /* Create a new window */
    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);
    gtk_window_set_title (GTK_WINDOW (window), "Check Button");

    /* Connect destroy event to the window. */
    gtk_signal_connect (GTK_OBJECT (window), "destroy",
                        GTK_SIGNAL_FUNC(gtk_main_quit), NULL);

    /* Creates a check button */
    button=gtk_check_button_new_with_label("Sono un check button");

    /* Add the button to window */
    gtk_container_add(GTK_CONTAINER(window),button);

    /* Connect "toggled" event to the button */
    gtk_signal_connect (GTK_OBJECT (button), "toggled",
                        GTK_SIGNAL_FUNC(togg), (gpointer *)button);
    gtk_widget_show(button);
    gtk_widget_show (window);

    gtk_main ();
    return(0);
}
```

Label (Etichette)

Le label permettono di inserire testo ovunque in una finestra.
Per creare una label si usa la funzione:

```
GtkWidget* gtk_label_new(const gchar *text);
```

Con la funzione

```
gtk_label_set_text(GtkLabel *label, gchar *text);
```

potete cambiare la stringa sull'etichetta in qualsiasi momento.

```
gtk_label_set_justify(GtkLabel *label, GtkJustification jtype);
```

La funzione gtk_label_set_justify è usata per giustificare il testo nell'etichetta. *jtype* può essere

- GTK_JUSTIFY_LEFT per posizionare il testo a sinistra,
- GTK_JUSTIFY_RIGHT per posizionare il testo a destra,
- GTK_JUSTIFY_CENTER per centrarlo,
- GTK_JUSTIFY_FILL per fare in modo che copra l'intera etichetta

```
gtk_label_set_line_wrap (GtkLabel *label,gboolean wrap);
```

è usato per permettere di spezzare il testo quando è più lungo dello spazio che dovrebbe contenerlo. Quando *wrap* è 1, il testo andrà a capo sulla prossima linea, altrimenti no.

```
gtk_label_get(GtkLabel *label, gchar **str)
```

viene usata per ricavare il testo contenuto in un'etichetta e metterlo nella variabile *str*.

ToolTip

I Tooltip sono testi che appaiono quando il mouse si ferma su un widget. Per esempio potete impostare un "tip" per un pulsante che dica "spedisci le informazioni" quando il mouse si ferma sopra il pulsante.

Per farlo bisogna creare un widget GtkToolTips prima:

```
GtkToolTips* gtk_tooltips_new();
```

E quindi attaccarlo a un widget:

```
gtk_tooltips_set_tip(GtkTooltips *tooltips, GtkWidget *widget,
                     const gchar *tip_text,const gchar *tip_private);
```

Un piccolo esempio:

```
#include <gtk/gtk.h>
int main( int argc,char *argv[] )
{
    GtkWidget *window;
    GtkWidget *button;
    GtkTooltips *tip;

    gtk_init (&argc, &argv);

    /* Create a new window */
    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);
    gtk_window_set_title (GTK_WINDOW (window), "Tooltips");
```

```
    /* Connect destroy event to the window. */
    gtk_signal_connect (GTK_OBJECT (window), "destroy",
                        GTK_SIGNAL_FUNC (gtk_main_quit), NULL);

    /* Creates a button */
    button=gtk_button_new_with_label("Sono un pulsante");

    /* Add the button to window */
    gtk_container_add(GTK_CONTAINER(window),button);

    /* Creates a tooltips*/
    tip=gtk_tooltips_new();

    /* Attache this tooltips to button with text*/
    gtk_tooltips_set_tip(tip, button, "Clickami!",NULL);

    gtk_widget_show(button);
    gtk_widget_show (window);

    gtk_main ();
    return(0);
}
```

Qualche altra funzione:

```
gtk_tooltips_enable (GtkTooltips *tooltips);
```

Attiva il tooltip.

```
gtk_tooltips_disable (GtkTooltips *tooltips);
```

Disattiva il tooltip.

Per ricavare i dati del tooltip di qualsiasi widget abbiamo bisogno di

```
GtkTooltipsData* gtk_tooltips_get_data(GtkWidget *widget);
```

GtkTooltipsData è una struct fatta in questo modo:

```
struct _GtkTooltipsData
{
   GtkTooltips *tooltips;
   GtkWidget *widget;
   gchar *tip_text;
   gchar *tip_private;
   GdkFont *font;
   gint width;
   GList *row;
};
```

Per impostare il ritardo di apparizione del testo,

```
gtk_tooltips_set_delay (GtkTooltips *tip, guint delay)
```

Combo Box

Una combo box è un campo di testo modificabile combinato con un menu a tendina. Si può inserire un valore o sceglierlo dal menu.

Si può creare una combo box con

```
GtkWidget *gtk_combo_new();
```

E ci serve una lista di opzioni sotto forma di struct GList.

```
GList *glist=NULL;
```

Si devono aggiungere le opzioni alla lista con

```
GList *g_list_append(GList *list, gchar *option);
```

Quindi si aggiunge la lista alla combo box

```
gtk_combo_set_popdown_strings(GtkCombo *combo, GList *List);
```

La combo box è pronta. Per leggere l'opzione selezionata usate:

```
gchar *gtk_entry_get_text(GtkEntry *entry);
```

entry è GTK_ENTRY(GTK_COMBO(combo)–>entry)) in questo caso.

```
gtk_combo_set_use_arrows(GtkCombo *combo,gint val);
```

Questa funzione viene usata per abilitare/disabilitare i cursori su/giù sulla tastiera per cambiare i valori nella combo box. Quando *val* è 0, questi non funzionano, altrimenti permettono di cambiare il valore. Ma quando il valore della combo è diverso da uno dei valori della lista, **i tasti non funzionano**.

```
gtk_combo_set_use_arrows_always(GtkCombo *combo,gint val);
```

Questa funzione è uguale a *gtk_combo_set_use_arrows* ma quando il valore della combo è diverso dai valori della lista **i tasti funzionano**.

```
gtk_combo_set_value_in_list(GtkCombo *combo, gboolean val,
                            gboolean ok_if_empty);
```

Quando *val* è 1, potete inserire un valore nella lista. Quando *ok_if_empty* è 1, il valore può essere vuoto.

```
#include <gtk/gtk.h>
void act(GtkWidget *widget, gpointer *data){
   g_print((gchar *)data);
}

int main( int argc,char *argv[] ) {
   GtkWidget *window;
   GtkWidget *combo;
   GtkWidget *button;
   GtkWidget *box;
   GList *list=NULL;
```

```
list=g_list_append(list,"Slackware");
list=g_list_append(list,"RedHat");
list=g_list_append(list,"SuSE");

gtk_init (&argc, &argv);

/* Create a new window */
window = gtk_window_new (GTK_WINDOW_TOPLEVEL);
gtk_window_set_title (GTK_WINDOW (window), "Combo Box");

/* Connect destroy event to the window. */
gtk_signal_connect (GTK_OBJECT (window), "destroy",
                    GTK_SIGNAL_FUNC(gtk_main_quit), NULL);

/* Create a new horizontal box */
box=gtk_hbox_new(1,0);
gtk_container_add(GTK_CONTAINER(window),box);

/* Creates a combo box */
combo=gtk_combo_new();

/* Sets the list */
gtk_combo_set_popdown_strings(GTK_COMBO(combo),list);

/* Enables up/down keys change the value. */
gtk_combo_set_use_arrows_always(GTK_COMBO(combo),1);

gtk_box_pack_start(GTK_BOX(box), combo,1,1,1);

button=gtk_button_new_with_label("Scrivi");
gtk_signal_connect(GTK_OBJECT(button),"clicked",GTK_SIGNAL_FUNC(act),
          gtk_entry_get_text(GTK_ENTRY(GTK_COMBO(combo)->entry)));
gtk_box_pack_start(GTK_BOX(box), button,1,1,1);

gtk_widget_show(box);
gtk_widget_show(combo);
gtk_widget_show(button);
gtk_widget_show (window);

gtk_main ();
return(0);
}
```

Ogni commento è gradito.
