GUI–Programmierung mit GTK – Teil 2

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

About the author:

Ich benutze Linux seit 1997. Freiheit, Flexibilität, Open Source. Diese Eigenschaften mag ich.

Translated to English by:
Özcan Güngör
<ozcangungor(at)netscape.net>

Abstract:

In diesem Artikel werden wir Boxen und Tabellen besprechen. Damit werden wir in der Lage sein, Komponenten in einer geeigneten Ordnung in Fenstern abzulegen. Um den Artikel zu verstehen, solltest du folgendes in der Programmiersprache C kennen:

- Variablen
- Funktionen
- Zeiger

Und zusätzlich ist es empfehlenswert, den vorherigen Artikel zu lesen.

Komponenten in Boxen verpacken

Komponenten zu verpacken bedeutet, sie in einer geeigneten Ordnung in Fenstern abzulegen. Eine der Möglichkeiten, dies in GTK zu bewerkstelligen, ist die Verwendung von Boxen. Der Grundgedanke dahinter ist, die Komponenten vertikal oder horizontal geordnet darin abzulegen. Also gibt es zwei Arten von Boxen: horizontale und vertikale. Ich will das erläutern.

Horizontale Boxen

In dieser Art von Box werden die Komponenten horizontal verpackt. Mit folgender Funktion wird eine horizontale Box erzeugt.

```
gtk_widget *box;
```

```
box=gtk_hbox_new(gboolean homogenous, gint spacing);
```

Dabei legt der Parameter *homogenous* fest, ob die Komponenten gleichmäßig verteilt werden oder nicht: ist er **TRUE**, füllen sie die Box so aus, daß der Abstand zwischen ihnen gleich ist; ist er **FALSE**, werden die Komponenten nebeneinander gelegt. *spacing* gibt einen Minimalabstand an.

Vertikale Boxen

Hier werden die Komponenten vertikal verpackt. Mit folgender Funktion wird eine vertikale Box erzeugt:

```
gtk_widget *box;
box=gtk_vbox_new(gboolean homogenous, gint spacing);
```

Die Parameter haben dieselbe Bedeutung wie in der vorigen Funktion.

Typische Beispiele für Boxen

Das Hinzufügen von Komponenten kann auf zwei Weisen geschehen. Die erste ist:

```
gtk_box_pack_start(GtkBox *box,GtkWidget *child,
                   gboolean expand, gboolean fill, guint padding);
```

Damit können wir Komponenten in eine Box legen (an die linke Seite einer horizontalen Box oder auf die obere Seite einer vertikalen). *box* ist die Box, der wir etwas hinzufügen wollen. *child* ist die hinzuzufügende Komponente. *expand* vergrößert die Komponente, so daß sie den ganzen Platz nutzt, wenn möglich. *padding* fügt zusätzlichen Platz links und rechts hinzu.

Das Gegenstück zu `gtk_box_pack_start` ist `gtk_box_pack_end`:

```
gtk_box_pack_end(GtkBox *box,GtkWidget *child,
                 gboolean expand, gboolean fill, guint padding);
```

So können wir Komponenten am Ende (rechts oder unten) einer Box hineinlegen. Die Parameter haben dieselbe Bedeutung wie zuvor.

Um eine Box einem Fenster hinzuzufügen, benutzen wir folgende Funktion:

```
gtk_container_add (GtkContainer *container,GtkWidget *component);
```

container ist das Fenster, dem die Box hinzugefügt wird, *component* ist die entsprechende Box. Um z.B. die oben erzeugte Box zu *window* hinzuzufügen, benutzen wir diese Funktion:

```
gtk_container_add(GTK_CONTAINER(window),box);

gtk_box_set_homogeneous (GtkBox *box, gboolean homogenous);
gtk_box_set_spacing(GtkBox *box, gint spacing);
```

Die erste Funktion im zweiten Kasten ändert die Eigenschaft *homogenous* (Gleichmäßigkeit) einer Box, und die zweite Funktion ändert ihren Platzbedarf. *box* ist die jeweils zu ändernde Box.

```
gtk_box_set_child_packing(GkBox *box,GtkWidget *shild,
       gboolean expand, gboolean  fill, guint padding,
       GtkPackType packingtype);
```

Diese Funktion redefiniert die Eigenschaften einer bereits verpackten Komponente. Die Parameter haben dieselbe Bedeutung wie in der Funktion `gtk_box_pack_start`. *packingtype* kann `GTK_PACK_START` oder `GTK_PACK_END` sein. Mit `GTK_PACK_START` wird die Komponente an den Boxanfang gelegt, wenn sie mit `gtk_pack_end` verpackt wurde. Mit `GTK_PACK_END` wird sie ans Boxende gelegt, wenn sie mit `gtk_pack_start` verpackt wurde.

Um das hier Erklärte besser zu verstehen, kannst du dir kutular.c ansehen.

Tabellen

Wie in HTML helfen uns Tabellen, Komponenten in Zellen zu fixieren. Dafür reicht es aus, eine Tabelle mit genügend Zeilen und Spalten zu erzeugen. Dann können wir eine Komponente in einer Zelle oder einer Gruppe von nebeneinanderliegenden Zellen unterbringen. Um eine Tabelle zu erzeugen, benutzen wir folgende Funktion:

```
GtkWidget *table;
GtkWidget* gtk_table_new(guint  row, guint column, gboolean homogenous);
```

row ist die Zahl der Zeilen, *column* die der Spalten. *homogenous* wird benutzt, um die Komponenten gleichmäßig zu verteilen.

Mit dieser Funktion fügen wir eine Komponente einer Tabelle hinzu:

```
void gtk_table_attach (GtkTable *table, GtkWidget *child,
         guint left_attach, guint right_attach, guint top_attach,
         guint bottom_attach, GtkAttachOptions xoptions,
         GtkAttachOptions yoptions, guint xpadding, guint ypadding);
```

table ist die Tabelle und *child* die darin einzufügende Komponente. *left_attach* ist die Nummer der Zelle, von links gezählt, in der sich die Komponente befinden soll, *right_attach* ebenso, von rechts gezählt, und *top_attach* bzw. *bottom_attach* analog von oben bzw. unten gezählt. Komponenten können mehr als eine Zelle bedecken.

xoptions und *yoptions* können drei verschiedene Werte annehmen: `GTK_FILL`, `GTK_EXPAND` oder `GTK_SHRINK`. `GTK_FILL` sorgt dafür, daß die Komponente sämtliche Zellen ausfüllt. `GTK_EXPAND` ordnet sie im Mittelpunkt der Zellen an, und `GTK_SHRINK` läßt sie in die Dimensionen der Zelle schrumpfen, wenn sie größer war. *xoptions* bewirkt dies nur auf der X−Achse, und *yoptions* nur auf der Y−Achse.

xpadding fügt Platz auf der linken und rechten Seite der Komponente hinzu (also entlang der X−Achse), während *ypadding* es entlang der Y−Achse macht.

Hier ist ein Beispielcode:

```
#include <gtk/gtk.h>

void delete_event( GtkWidget *widget,GdkEvent  *event,gpointer data )
{
    gtk_main_quit ();
}
```

```
int main( int   argc,char *argv[] )
{
    GtkWidget *window;
    GtkWidget *button;
    GtkWidget *table;

    gtk_init (&argc, &argv);

    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);

    gtk_signal_connect (GTK_OBJECT (window), "delete_event",
                        GTK_SIGNAL_FUNC (delete_event), NULL);

    table = gtk_table_new (2, 2, TRUE);

    gtk_container_add (GTK_CONTAINER (window), table);

    button = gtk_button_new_with_label ("button 1");
    gtk_table_attach(GTK_TABLE(table), button, 0, 1, 0, 2,GTK_SHRINK,
                     GTK_SHRINK,0,0);
    gtk_widget_show (button);

    button = gtk_button_new_with_label ("button 2");
    gtk_table_attach (GTK_TABLE(table), button, 1, 2, 1, 2,
                      GTK_SHRINK,GTK_SHRINK,0,0);
    gtk_widget_show (button);

    button = gtk_button_new_with_label ("button 3");
    gtk_table_attach (GTK_TABLE(table), button, 1, 2, 0, 1,
                      GTK_SHRINK,GTK_SHRINK,0,0);
    gtk_widget_show (button);

    gtk_widget_show (table);
    gtk_widget_show (window);

    gtk_main ();

    return 0;
}
```

Weil `gtk_table_attach` eine Menge Parameter hat, wird eine neue, kurze Funktion erzeugt: `gtk_table_attach_defaults`. Sie erfüllt dieselbe Aufgabe, nur mit weniger Parametern.

```
void gtk_table_attach_defaults (GtkTable *table,GtkWidget *child,
                guint left_attach, guint right_attach, guint top_attach,
                guint bottom_attach);
```

Hier haben die Parameter dieselbe Bedeutung. *xoptions* und *yoptions* haben den Wert `GTK_FILL|GTK_EXPAND`. *xpadding* und *ypadding* haben den Wert 0.

Mit dieser Funktion wird die Zahl der Zeilen und Spalten einer bestehenden Tabelle geändert:

```
void gtk_table_resize(GtkTable *table, guint rows, guint columns);
```

Mit folgenden Funktionen kannst du die Werte für die Abstände einer Zeile oder einer Spalte ändern:

```
void gtk_table_set_row_spacing (GtkTable *table, guint row,
                                guint spacing);
void gtk_table_set_col_spacing (GtkTable *table, guint column,
                                guint spacing);
```

Diese ändern die Abstandswerte ganzer Zeilen oder Spalten:

```
void gtk_table_set_row_spacings (GtkTable *table, guint spacing);
void gtk_table_set_col_spacings (GtkTable *table, guint spacing);
```

Und diese ändern schließlich die Gleichmäßigkeits−Schalter einer bestehenden Tabelle:

```
void gtk_table_set_homogeneous (GtkTable *table, gboolean homogenous);
```

Fazit

In diesem Artikel haben wir die Methoden kennengelernt, um Komponenten zu verpacken, und dann haben wir uns Funktionen angesehen, um einige Eigenschaften von Boxen und Tabellen festzulegen. Ich freue mich immer über Fragen, Kommentare und Ideen von Lesern. Schick mir einfach eine e−mail...
