LinuxFocus article number 303
http://linuxfocus.org

Grafisch programmeren met GTK - Deel 2

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

Over de auteur:
Sinds 1997 gebruik ik Linux. Vrijheid, flexibiliteit en open broncode. Dat zijn de eigenschappen die mijn hart hebben veroverd.

Vertaald naar het Nederlands door:
Samuel Derous
<cyberprophet/at/linux.be>

Kort:

In dit artikel zullen we het hebben over kaders en tabellen. Hiermee zullen we componenten in een bepaalde volgorde op vensters kunnen instellen. Om deze artikelen te begrijpen zou je de volgende dingen van de C programmeertaal moeten kennen:

- Variabelen
- functies
- Pointers

En natuurlijk is het aangeraden om ook het vorige artikel te lezen.

Het inkapselen van componenten met kaders

Met het inkapselen van componenten wordt bedoeld dat de componenten in een bepaalde volgorde op een venster geplaatst kunnen worden. Eén van de manieren om dit in GTK te doen, is om gebruik te maken van kaders. De hoofd-idee achter de kaders is om de componenten in een verticale of horizontale volgorde in te kapselen. Er zijn twee soorten kaders: horizontale en verticale. Laten we deze types uitleggen:

Horizontale kaders

In dit kadertype, worden componenten horizontaal gegroepeerd. De volgende functie wordt gebruikt om

een horizontaal kader te maken.

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

Waar de parameter homogenous wordt gebruikt om te definiëren of de componenten homogeen verspreid zullen worden, of niet: Als dit TRUE is dan zullen de componenten de gehele doos vullen zodat er een gelijke afstand tussen hen is en als het FALSE is worden de componenten naast elkaar verpakt. Spacing wordt gebruikt om de minimum ruimte tussen de componenten vast te stellen.

Verticale kaders

In dit kadertype worden componenten verticaal verpakt. De volgende functie wordt gebruikt om een verticaal kader aan te maken:

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

De parameters van deze functie hebben dezelfde betekenis als deze in de functie voor horizontale kaders.

Algemene eigenschappen van kaders

Er zijn twee manieren om componenten toe te voegen. De eerste is:

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

Door gebruik te maken van deze functie kunnen we componenten in een kader toevoegen (aan de linkerzijde voor een horizontaal kader, en bovenaan voor een verticaal kader). Box is het kader waar we een component aan toe willen voegen. Child is het component dat toegevoegd moet worden. Expand wordt gebruikt om de grootte van het component uit te breiden zodat het alle mogelijke ruimte gebruikt. Padding wordt dan weer gebruikt om bijkomende ruimte aan de linker- en rechterzijde toe te voegen.

De bijkomende functie voor gtk_box_pack_start is gtk_box_pack_end:

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

Deze functie laat ons toe om componenten aan het einde van het kader toe te voegen (aan de rechterzijde, of beneden). De parameters hebben dezelfde betekenis als in de voorgaande functie.

De volgende functie wordt gebruikt om een kader aan een venster toe te voegen:

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

container is het venster waar het kader aan toegevoegd zal worden, component is het kader die toegevoegd moet worden. Om b.v. het kader dat we hierboven aangemaakt hebben aan een venster toe te voegen, wordt volgend commando gebruikt:

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

De eerste functie hierboven wordt gebruikt om de homogene eigenschap van een kader aan te passen en de tweede functie om de grootte van de ruimte van een kader te veranderen.

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

Deze functie herdefiniëert de eigenschappen van een component die al verpakt is. De parameters hebben dezelfde betekenis als in de functie gtk_box_pack_start. packingtype kan zowel GTK_PACK_START of GTK_PACK_END zijn. GTK_PACK_START zorgt ervoor dat het component aan het begin van een kader wordt verpakt als het component gebruik maakt van de functie gtk_pack_end. GTK_PACK_END zorgt ervoor dat een component aan het einde van een kader wordt verpakt als het component verpakt wordt door gebruik te maken van de functie gtk_pack_start.

Om beter te begrijpen wat hier net uitgelegd werd, probeer kutular.c.

Tabellen

Tabellen helpen ons, net als in HTML, om componenten op te maken in cellen. Om dit te bewerkstelligen is het voldoende om een tabel met genoeg rijen en kolommen te creëren. Dan kunnen we een component lokaliseren in een cel of in een groep cellen (wat enkel toegestaan is voor cellen die zich zij aan zij bevinden). Gebruik volgende functie om een tabel aan te maken:

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

row is het aantal rijen, column het aantal kolommen. homogenous wordt gebruikt om de componenten homogeen te verspreiden.

De volgende functie wordt gebruikt om een component aan een tabel toe te voegen:

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

table stelt de tabel voor waar de componenten aan toegevoegd zullen worden en child is het component die toegevoegd zal worden. left-attach betreft het nummer van de cel aan de linkerzijde waar het component in gevoegd zal worden. right-attach is het nummer van de cel aan de rechterzijde waar het component in gevoegd zal worden. top-attach is het numer van de cel aan de bovenkant waar het component in geplaatst zal worden en bottom-attach is het numer van de cel aan het einde waarin het

component zal geplaatst worden.
Componenten kunen meer dan één cel beslaan.

xoptions en yoptions kunnen drie verschillende waarden krijgen: GTK_FILL, GTK_EXPAND, GTK_SHRINK. GTK_FILL zorgt ervoor dat het component de volledige cel vult. GTK_EXPAND zorgt ervoor dat het component gecentreerd wordt in de cel en GTK_SHRINK zorgt ervoor dat het component passend gemaakt wordt voor de cel(len) als het component groter dan de cel(len) zou zijn. xoptions maakt deze veranderingen enkel op de x-as en yoptions enkel op de y-as.

xpadding zet ruimte aan de linker- en rechterzijde van het component aan de x-as en ypadding doet hetzelfde aan de y-as.

Hier is een voorbeeldcode:

```
#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;
}
```

Daar gtk_table attach heel wat paramaters heeft, wordt een nieuwe korte functie gemaakt: gtk_table_attach_defaults. Deze functie doet net hetzelfde maar met minder paramaters.

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

De paramaters hebben hier dezelfde betekenis. xoptions en yoptions hebben de waarde GTK_FILL en GTK_EXPAND. xpadding en ypadding hebben de waarde 0.

De volgende functie wordt gebruikt om het aantal rijen en kolommen van een bestaande tabel aan te passen:

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

Door gebruik te maken van volgende functies, kan je de waarde van de ruimte van een rij of kolom aanpassen:

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

De volgende functie past de ruimte van hele rijen of kolommen aan:

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

De volgende functie past de homogene waarde van een bestaande tabel aan:

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

Samenvatting

In dit artikel hebben we geleerd hoe we componenten kunnen verpakken, en daarna hebben we een kijkje genomen naar functies voor enkele eigenschappen van kaders en tabellen. Ik ontvang graag vragen, commentaar en ideeën van lezers. Zend me dus gewoon een e-mail...

Vertaling info:
tr --> -- : Özcan Güngör <ozcangungor(at)netscape.net>
en --> tr: Özcan Güngör <ozcangungor(at)netscape.net>
en --> nl: Samuel Derous <cyberprophet/at/linux.be>

2005-01-14, generated by lfparser_pdf version 2.51