Linux图形界面编程

❖什么是X Window系统

- X Window是UNIX和Linux系统上的图形界面系统
- X Window是众多软件程序的组合体，是一个程序库
- X Window是内核之上的一个应用

❖X Window系统的组成

- X Server:控制输出及输入设备并维护相关资源的程序
- X Client:应用程序的核心部分，与硬件无关，每个应用程序都是一个X Client
- X协议:X Server与X Client间的通信协议
- Window Manager:提供窗口控制参数，包括窗口大小、重迭显示、移动、最小化等
- Display Manager:管理登录界面

❖什么是GTK

- GTK (GIMP Toolkit) 是一套用于创建图形用户界面的工具包。
- GTK 实质上是一个面向对象的应用程序接口 (API)。尽管完全用C写成的，但它是基于类和回调函数 (指向函数的指针) 的思想实现的。
- 它遵循LGPL许可证，所以可以用它来开发开源软件、自由软件，甚至是封闭源代码的商业软件，而不用花费任何费用来购买许可证和使用权。
- 当前，GTK已经被成功地应用到了大量的自由软件及商业软件中，已经取得了很大的成功。

❖GTK构成

- Glib：包含一些标准函数的替代函数，以及一些处理链表等数据结构的函数等。这些替代函数被用来增强GTK 的可移植性，同时提供 libc 的对应函数的增强版本。
- Pango：用来处理国际化文字输出。
- GDK ：为GTK提供底层实现的函数库。
- GdkPixbuf ：用于加载和维护图像“缓存”的函数库。
- Cairo ：用于制作二维图像的函数库。

Glib的数据类型

Glib具有一套自己的类型系统，与C语言标准类型对照如下

C类型	GLIB类型	C类型	GLIB类型
char	gchar	unsigned char	guchar
short	gshort	unsigned short	gushort
long	glong	unsigned long	gulong
int	gint	unsigned int	guint
int	gboolean	void *	gpointer
float	gfloat	const void *	gconstpointer
double	gdouble		

GTK的数据类型

- Gtk+中采用了面向对象的概念，通常由构件派生构件。

  如铵钮构件（GtkButton）由容器构件（GtkContainer）派生；容器构件由通用构件（GtkWidget）派生；通用构件又由GtkObject派生。

- 所有建立构件的函数返回指向GtkWidget的指针。

  比如gtk_window_new返回的是GtkWidget * 而不是GtkWindow *。这使得通用函数可以对所有的构件进行操作。

- 在调用具体的构件函数之前将构件转换为正确的类型。

  如：

  void gtk_window_set_title (GtkWindow * window, const gchar *title);

  第一个参数要求为一个GtkWindow *类型。

- 每一种构件有一个转换宏可将GtkWidget转换为相应构件类型。

  如：

  gtk_window_set_title(GTK_WINDOW(window), “helloGTK”);

❖界面基本元素

1、窗口

窗口是一个应用程序的界面框架，程序的所有内容和与用户的交互都在这个窗口中。在设置应用程序的界面时，第一步便是建立一个窗口。

- 新建一个窗口：

```
#include <gtk/gtk.h>
GtkWidget *gtk_window_new (GtkWindowType type);
```

type:GTK-WINDOW-TOPLEVEL

GTK-WINDOW-POPUP

成功:返回一个GtkWidget类型的指针，

失败:返回空指针。

gtk_window_new 函数根据给出的窗口类型创建窗口，默认情况下窗口为200X200像素大小。

- 显示窗口：

当新建一个窗口后，这个窗口不会马上就显示出来，需要调用窗口显示函数gtk_widget_show()来显示这个窗口，函数原型如下：

```
#include <gtk/gtk.h>
void gtk_widget_show(GtkWidget * widget);
```

参数widget是一个GtkWidget类型的结构体

- 设置标题：

gtk_window_set_title()函数用于设置窗口的标题，函数的原型如下：

```
#include <gtk/gtk.h>
gtk_window_set_title(GTK_WINDOW *window,gchar *title);
```

window 表示将要设置标题的窗口构件，

title表示设置的标题，函数无返回值。

注意title的值要是英文否则在有些Linux下会乱码。

- 设置窗口的大小和位置：

```
gtk_widget_set_usize(GtkWidget * widget,int x,int y);
gtk_widget_set_uposition(GtkWidget * widget,int x,int y);
```

实例演示：

```
#include <gtk/gtk.h>
int main(int argc,char *argv[])
{
    GtkWidget *window;
    char title[]=  “My First Window";
    gtk_init(&argc,&argv);
    window = gtk_window_new(GTK_WINDOW_TOPLEVEL);

    gtk_window_set_title(GTK_WINDOW (window),title);
    gtk_widget_set_usize((window),400,200);
    gtk_widget_set_uposition((window),200,200);
    gtk_widget_show(window);
    gtk_main();
    return 0;
}
```

编译方法：

gcc hellogtk.c -o hellogtk `pkg-config --cflags --libs gtk+-2.0`

注意：

（`）不是普通的单引号（’），而是同"~"在一起的那个符号！

2、按钮

在图形界面的程序中，有很多操作都是通过窗口程序的按钮来实现的。在后面我们还将看到，按钮最常用于发送一个信号，这个信号会引起相应事件的响应。

- 新建一个按钮：

```
#include <gtk/gtk.h>
GtkWidget *gtk_button_new_with_lable (gchar *lable);
```

若创建成功则返回GtkWidget类型的指针，否则返回NULL

- 设置和获取按钮的标签：

```
#include <gtk/gtk.h>
const gchar *gtk_button_get_lable (GtkButton *button);
void gtk_button_set_lable(GtkButton *button,const
                                        gchar *lable);
```

3、文本框

文本框是界面的输入区域，用户可以在这个区域中用键盘输入内容，界面程序的各种输入都是通过文本框来完成的。

- 新建一个文本框:

```
#include <gtk/gtk.h>
GtkWidget *gtk_entry_new(void);
```

若创建成功则返回一个GtkWidget类型的指针，若失败返回NULL。

另建一个文本框的函数为：

```
GtkWidget *gtk_entry_new_with_man_length(gint max);
```

返回值同上，参数max是表示该文本框最多可以输入的字符。

- 设置和获取文本框的数据

```
#include <gtk/gtk.h>
const gchar *gtk_entry_get_text(GtkEntry *entry);
void gtk_entry_set_text(GtkEntry *entry,const gchar *text);
```

在文本框输入数据后往往要获得数据和进行相关的处理。函数gtk_entry_get_text()用来获得数据。gtk_entry_set_text()用来设置文本框的初始内容。

第一个函数返回值：若成功则返回指向文本框中的字符串的指针，若失败则返回NULL。

第二个函数无返回值。

在参数列表中，entry是一个指向文本框的指针，text表示需要设置文本框中的字符串文本。

实例演示：button.c

GTK界面布局

GTK+的图形界面编程中的界面布局构件，包括表格，框，窗格等，其中表格是界面编程中最常使用的布局构件，通过在表格的单元格中插入不同的构件，来实现构件的布局和排列。使用界面布局构件，可以在一个窗口中设计出复杂而优美的界面。

❖ GTK界面布局

（1）表格

表格是指用横竖布局的线和格子将一个窗口划分成多个区域，每个区域可以放置不同的构件。如果一个构件中可以存放其它的构件，这个构件杯称做容器。GTK+的容器都是二进制的，也就是每个容器只能放置一个构件，如果想在一个窗口中放置多个构件，则需要使用表格，窗格等有多个单元格的容器。

- 表格的建立

```
#include <gtk/gtk.h>
GtkWidget *gtk_table_new(guint rows,
                         guint columns,
                         gboolean homegenous)
```

rows表示行数，columns表示列数；

homegenous是一个布尔值，

设为true，每个单元格的大小相同。

设为false，根据单元格中的构件大小自行调整。

注意：表格的作用只是将窗口划分成不同的区域，本身并不显示，也就是说在GTK中，表格作为一个容纳其它构件的容器，并不会实际显示。

- 添加构件到表格中

```
#include <gtk/gtk.h>
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：容器表格指针；

child：需要添加的构件的指针；

left_attach，right_attach：表示左右是表格的第几条边（从0计数）；

top_attach，bottom_attach：表示上下是表格的第几条边；

xoptions，yoptions：表示水平和垂直方向对齐；

xpading，ypading：表示构件与边框在水平和垂直放向上的边距；

- 嵌套表格

在设计复杂的界面时，使用一个表格并不能完成布局，这时就需要在表格中嵌套表格，表格也是一个普通的构件，可以把表格添加到另一个表格的单元格之中，这样通过表格的嵌套就可以实现复杂的布局了。

在下面的这个例子中实现表格的综合应用，它包括建立表格，合并单元格，以及嵌套表格。

实例演示：table.c

（2）框

在GTK+中框是一种不可见的widget容器，它有水平框和垂直框两种。水平框是指构件放入窗口的顺序水平排列。垂直框是指构件按放入窗口的顺序垂直排列。水平框可以看做是只有一行的表格，而垂直框可以看做是只有一列的表格。但是它们的操作比表格简单，放置构件时不需要考虑构件的位置。

- 框的建立

```
#include <gtk/gtk.h>
GtkWidget *gtk_hbox_new (gboolean homogeneous,
                         gint spacing);
GtkWidget *gtk_vbox_new (gboolean homogeneous,
                         gint spacing);
```

两个函数的返回：如成功则返回一个GtkWidget类型的指针，失败则返回NULL

homogeneous表示放入框中的构件是否具有相同的高度或宽度。

spacing表示每一行构件之间的距离。

同样的，建立一个框后，需要用gtk_container_add()函数来将这个框添加到窗口中，并且调用显示函数gtk_widget_show()来显示这个框。

- 在框中添加构件

```
#include <gtk/gtk.h>
void gtk_box_pack_start(GtkBox *box,GtkWidget    *child,
                        gboolean expand,gboolean fill,
                        guint padding);
void gtk_box_pack_end(GtkBox *box,GtkWidget *child,
                        gboolean expand,gboolean fill,
                        guint padding);
```

将构件放入框容器中，前者从左到右，从上到下，后者相反。

同表格一样，框也是一个容器，当没有向这个容器中添加任何构件时，容器是不能显示的。

（3）窗格

窗格也是GTK+图形界面编程中常用的布局方式之一，它可以把一个界面划分成水平或者垂直的两个区域，拖动两个区域的分界线，可以改变两个窗格的大小。窗格有水平窗格和垂直窗格两种。

- 创建窗格：

```
#include <gtk/gtk.h>
GtkWidget *gtk_hpaned_new(void);  //水平窗格
GtkWidget *gtk_vpaned_new(void);  //垂直窗格
```

成功则返回一个GtkWidget类型的指针，若失败则返回NULL。

- 在窗格中添加构件

```
#include <gtk/gtk.h>
void gtk_paned_pack1(GtkPaned,GtkWidget *child,
                     gboolean resize,gboolean shrink);
void gtk_paned_pack2(GtkPaned *paned,GtkWidget *child,
                     gboolean resize,gboolean shrink);
```

作为GTK+的容器，在没有向窗格中添加任何构件时，窗格是不能显示的。

GTK+其它常用构件

GTK+的其它常用构件包括进度条，微调按钮，组合框，单选按钮，复选按钮，下拉菜单，对话框等。

❖信号与回调函数

在以前的例子中我们点击提交按钮时，文本框的字符串信息会输出到终端中，这表明发生了一个单击事件。实际上单击按钮时系统自动调用了相关的事件响应函数。图形用户界面的程序是事件驱动的程序。程序进入gtk_main()函数后，等待事件的发生，一旦发生某个事件，相应的信号将产生。如果程序中定义了相应的消息处理函数（或称回调函数），系统会自动进行调用。

❖信号与回调函数

（1）添加信号：

```
#include <gtk/gtk.h>
gulong g_signal_connect (GtkObject *object,gchar *name
                         Gcallback callback_func,
                         gpointer func_data);
```

object：是一个构件的指针，指向将产生信号的构件；

name：表示消息或者事件的类型；

函数功能：把一个信号处理函数添加到一个构件上，在构件和消息处理函数间建立关联。

（2）事件类型：

activate : 激活的时候发生
clicked : 单击以后发生
enter : 鼠标指针进入这个按钮以后发生
leave : 鼠标离开按钮以后发生
pressed : 鼠标按下以后发生
released : 鼠标释放以后发生
destroy : 关闭窗口时发生

如下面这个例子表示鼠标点击以后发生：

```
g_signal_connect(G_OBJECT(button),“clicked”,
                        G_CALLBACK(on_clicked),window);
```

（3）回调函数：

```
#includ<gtk/gtk.h>
void callback_func(GtkWidget *widget,gpointer func_data);
```

widget：指向要接收消息的构件；

func_data：指向消息产生时传递给该函数的数据。

实例演示：destroy.c

Beyond Technology