**关闭屏幕超时设置**

**一、分析工作**

关闭屏幕超时设置任务需要实现的是，由gtk界面（客户端）设置超时时间，经由GDBus方式，控制manager(服务器端)完成判断超时时间是否到达，若到达，则调用关闭屏幕函数。

由于关闭客户端后，之前设置的参数会随着关闭一起消失，所以再次开启客户端时需要从服务器端获取之前设置的超时时间值。

1. 完成客户端的编程，功能是当客户端开启时，首先获取服务器端之前由客户端设置的超时时间值，若未设置过，则获取默认值；获取超时时间值后，将这个值设置到滑动栏中显示出来。
2. 客户端开启后，通过拖动滑动条设置超时时间值，滑动条的数值对应超时时间值，当设定好后，点击确定按钮，将设定的值“发送”给服务器端。
3. 完成服务器端的编程，功能是完成对客户端“发送”过来的值，与客户端要“获取”值时的操作。
4. **客户端编程实现**

#include <gtk/gtk.h>

#include <stdio.h>

#include "generated.h"

GtkBuilder \*builder = NULL;

GtkWidget \*dialog = NULL;

GtkWidget \*button = NULL;

GtkWidget \*hscale = NULL;

GtkAdjustment \*adj = NULL;

ItepPowerManagerDisplay \*display;

GError \*error = NULL;

gint gettime = 0;

gdouble settime = 0;

void on\_button\_clicked(GtkWidget \*widget, gpointer data)

{

/\*获取滑动条中的超时时间值\*/

settime = gtk\_adjustment\_get\_value(adj);

/\*客户端发送设置好的超时时间值到服务器端\*/

itep\_power\_manager\_display\_call\_set\_timeout\_sync(display, (gint)settime, NULL, NULL);

}

gint main(int argc, char \*argv[])

{

display = itep\_power\_manager\_display\_proxy\_new\_for\_bus\_sync(

G\_BUS\_TYPE\_SESSION,

G\_DBUS\_PROXY\_FLAGS\_NONE,

"cn.com.itep.powermanager.display",

"/cn/com/itep/powermanager/display",

NULL,

&error);

if (display == NULL)

{

g\_printf("Error getting proxy: %s\n",error->message);

g\_error\_free(error);

return 0;

}

gtk\_init(&argc,&argv);

builder = gtk\_builder\_new();

gtk\_builder\_add\_from\_file(builder,GLADE\_FILE,NULL);

button = GTK\_WIDGET(gtk\_builder\_get\_object(builder,"button1"));

dialog = GTK\_WIDGET(gtk\_builder\_get\_object(builder,"dialog1"));

hscale = GTK\_WIDGET(gtk\_builder\_get\_object(builder,"hscale1"));

adj = GTK\_ADJUSTMENT(gtk\_builder\_get\_object(builder,"adj"));

/\*客户端向服务器端申请获得超时时间值，保存在gettime中\*/

itep\_power\_manager\_display\_call\_get\_timeout\_sync(display, &gettime, NULL, NULL);

/\*将获得的超时时间值在滑动条中显示\*/

gtk\_adjustment\_set\_value(adj,(gdouble)gettime);

g\_signal\_connect(G\_OBJECT(button), "clicked",G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(G\_OBJECT(dialog), "destroy",G\_CALLBACK(gtk\_main\_quit), NULL);

gtk\_widget\_show\_all(dialog);

gtk\_main();

return 0;

}

1. **服务器端编程实现。**

#include <stdio.h>

#include <gtk/gtk.h>

#include "itep-power-manager.h"

///\*

#include "generated.h"

#define DEBUG\_LIUHW 0

//\*/

///\*

/\*设置savetimeout静态变量,用来保存设置的超时时间值\*/

static gint savetimeout = 0;

ItepPowerManager \*ipmtemp;

/\*客户端要获取服务器端的超时时间值时，从处理函数中获取超时值保存到savetimeout中，再将保存的savetimeout返回给客户端\*/

static gboolean display\_get\_timeout(ItepPowerManagerDisplay \*display,

GDBusMethodInvocation \*invocation)

{

savetimeout = itep\_power\_manager\_display\_get\_timeout(ipmtemp);

itep\_power\_manager\_display\_complete\_get\_timeout(display,invocation,savetimeout);

return TRUE;

}

/\*客户端发送超时时间值到服务器时，将发送的值保存到savetimeout中，同时设置给处理函数\*/

static gboolean display\_set\_timeout(ItepPowerManagerDisplay \*display,

GDBusMethodInvocation \*invocation,

gint timeout)

{

savetimeout = timeout;

itep\_power\_manager\_display\_set\_timeout(ipmtemp, savetimeout);

itep\_power\_manager\_display\_complete\_set\_timeout(display,invocation);

if (DEBUG\_LIUHW)

g\_printf("success set\_timeout = %d\n",savetimeout);

return TRUE;

}

static void on\_bus\_acquired(GDBusConnection \*connection,

const gchar \*name,

gpointer user\_data)

{

if (DEBUG\_LIUHW)

g\_printf("on\_bus\_acquired\n");

ItepPowerManagerDisplay \*display = NULL;

display = itep\_power\_manager\_display\_skeleton\_new();

/\*当客户端要获取服务器端的超时时间值时，执行display\_get\_timeout函数\*/

g\_signal\_connect(display,"handle-get-timeout",G\_CALLBACK(display\_get\_timeout),NULL);

/\*当客户端发送超时时间值到服务器时，执行display\_set\_timeout函数\*/

g\_signal\_connect(display,"handle-set-timeout",G\_CALLBACK(display\_set\_timeout),NULL);

g\_dbus\_interface\_skeleton\_export(G\_DBUS\_INTERFACE\_SKELETON(display),connection,"/cn/com/itep/powermanager/display",NULL);

}

static void on\_name\_acquired(GDBusConnection \*connect,

const gchar \*name,

gpointer user\_data)

{

if (DEBUG\_LIUHW)

g\_printf("Acquired the name %s\n",name);

}

static void on\_name\_lost(GDBusConnection \*connection,

const gchar \*name,

gpointer user\_data)

{

if (DEBUG\_LIUHW)

g\_printf("Lost the name %s\n",name);

}

//\*/

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

{

int timeout = 0;

ItepPowerManager \*ipm;

gtk\_init(&argc, &argv);

///\*

g\_bus\_own\_name(G\_BUS\_TYPE\_SESSION,

"cn.com.itep.powermanager.display",

G\_BUS\_NAME\_OWNER\_FLAGS\_ALLOW\_REPLACEMENT | G\_BUS\_NAME\_OWNER\_FLAGS\_REPLACE,

on\_bus\_acquired,

on\_name\_acquired,

on\_name\_lost,

NULL,

NULL);

//\*/

ipm = itep\_power\_manager\_new();

itep\_power\_manager\_start(ipm);

ipmtemp = ipm;

gtk\_main();

return 0;

}