บทที่ 7

ลักษณะของ Signal

การเขียนโปรแกรมด้วย GTK นั้น จะเป็นการเขียนโปรแกรมแบบ event driven นั่นก็คือเราจะต้องกำหนดหรือออกแบบโปรแกรมของเราก่อนว่าเราจะดักเหตุการณ์ (signal หรือ event) อะไรบ้าง โดยโปรแกรมจะรอรับเหตุการณ์ต่างๆ จากนั้นก็จะส่งไปยังฟังก์ชันการทำงานที่เรากำหนดไว้ต่อไป

ในการจัดการกับการเขียนโปรแกรมตามที่ได้กล่าวไปข้างต้นนั้นจะต้องอาศัยแนวความคิดที่เรียกว่า signal ซึ่งก็คือเหตุการณ์ต่างๆ ที่เกิดขึ้น เช่นการกดปุ่มปิดหน้าต่าง การใช้ดับเบิ้ลคลิ๊กเป็นต้น ในกรณีที่เกิด signal ขึ้นมาแล้วนั้นจะให้โปรแกรมเราทำอะไรต่อไปจะกำหนดไว้เป็น signal handler เพื่อที่จะดักจับ signal แล้วส่งการทำงานไปยังฟังก์ชันที่เตรียมเอาไว้ signal handler มีรูปแบบการใช้สองรูปแบบ

รูปแบบที่หนึ่งคือ

```
gulong  g_signal_connect ( gpointer        *object,
                           const gchar     *name,
                           Gcallback       function,
                           gpointer        data );
```

โดยที่อาร์กิวเมนต์ตัวแรกจะเป็น widget ที่ต้องการจะดักจับ signal ตัวที่สองจะเป็นชื่อของ signal ที่ใช้ในการดักจับ ตัวที่สามจะเป็นฟังก์ชันที่เรียกใช้เมื่อมี signal ที่ดักจับไว้เกิดขึ้น และอาร์กิวเมนต์ตัวสุดท้ายจะเป็นข้อมูลที่จะผ่านเข้าไปในฟังก์ชันที่เรียกใช้ในอาร์กิวเมนต์ตัวที่สาม

สำหรับฟังก์ชันที่เรียกใช้ในอาร์กิเมนต์ตัวที่สามนั้น เรียกว่า “callback function” โดยมีรูปแบบทั่วไปดังนี้

```
void  callback_function ( GtkWidget        *widget,
                          gpointer         data);
```

เมื่ออาร์กิวเมนต์ตัวแรกเป็นพอยต์เตอร์ที่ชี้ไปยัง widget ที่เกิด signal ขึ้น ส่วนตัวที่สองเป็นพอยต์เตอร์ที่ชี้ไปยังข้อมูลที่ส่งมาจากอาร์กิวเมนต์ตัวที่สี่ของ g_signal_connect

ในการเขียนโปรแกรมโดยทั่วไปที่เรากำหนดการทำงานของ callback function เองนั้นส่วนใหญ่จะใช้ signal handler รูปแบบที่หนึ่งที่ได้กล่าวมาแล้วนั่นเอง

รูปแบบที่สอง

```
gulong  g_signal_connect_swapped ( gpointer          *object,
                                   const gchar       *name,
                                   Gcallback         function,
                                   gpointer          *slot_object );
```

แบบนี้ก็จะเหมือนกับ g_signal_connect() ต่างกันตรงที่จำนวนอาร์กิวเมนต์ของ callback function ซึ่งมีเพียงหนึ่งอาร์กิวเมนต์และเป็นพอยต์เตอร์ที่ชี้ไปยัง GTK object รูปแบบของ callback function เป็นดังนี้

```
void  callback_function ( GtkObject        *object );
```

เมื่อ object คือ widget ที่ใช้งานบ่อยๆ

จุดประสงค์ของการมีฟังก์ชันที่ติดต่อกับ signal ไว้สองฟังก์ชันก็เพื่ออนุญาตให้ callback มีจำนวนอาร์กิวเมนต์ที่แตกต่างกัน หลายฟังก์ชันใน GTK library จะยอมรับเฉพาะ GtkWidget ที่ชี้ไปยังอาร์กิวเมนต์ตัวเดียว พูดง่ายก็ก็การใช้ g_signal_connect_swapped() จะเป็นการเรียกใช้ callback function ที่มีอยู่แล้วใน GTK library โดยที่ไม่ต้องเขียน callback function เอง

สำหรับเหตุการณ์ต่างๆ ที่เราดักในการเขียนโปรแกรมมีดังนี้

event	focus_out_event
button_press_event	map_event
button_release_event	unmap_event
motion_notify_event	property_notify_event
delete_event	selection_clear_event
destroy_event	selection_request_event
expose_event	selection_notify_event
key_press_event	proximity_in_event
key_release_event	proximity_out_event
enter_notify_event	drag_begin_event
leave_notify_event	drag_request_event
configure_event	drag_end_event
focus_in_event	drop_enter_event

drop_leave_event

drop_data_available_event

other_event

ตัวอย่างสำหรับการดักเหตุการณ์และ signal ดูจากตัวอย่างต่อไปนี้

Hello1.c

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

	gtk_init(&argc,&argv);

	/* create new window */
	window = gtk_window_new(GTK_WINDOW_TOPLEVEL);

	/* create button */
	button = gtk_button_new_with_lebel("Hello world");

	gtk_widget_show(button);
	gtk_widget_show(window);

	gtk_main();

	return(0);
}
```

จากโปรแกรม Hello1.c จะมีหน้าต่างและปุ่มชื่อ Hello world อยู่หนึ่งปุ่ม หากเราลองกดปุ่มปิดหน้าต่างของโปรแกรม หรือกดปุ่ม Hello world นั้นก็จะไม่มีอะไรเกิดขึ้นเนื่องจากว่าเรายังไม่ได้ทำการดักเหตุการณ์ใดๆที่จะเกิดนั่นเอง

ในตัวอย่างต่อไปจะเป็นการนำเอาโปรแกรม Hello.c มาดัดแปลงโดยเพิ่มส่วนของการดักเหตุการณ์ลงไปในโปรแกรม

Hello2.c

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

/* callback function */
void hello(GtkWidget *,gpointer data)
{
        g_print("%s\n" , (char *) data);  //print string
}

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

        gtk_init(&argc,&argv);

        /* create new window */
        window = gtk_window_new(GTK_WINDOW_TOPLEVEL);

        g_signal_connect(G_OBJECT(window) , "delete_event" ,
                        G_CALLBACK (gtk_main_quit) , NULL);
        /* เป็นการดักเหตุการณ์ delete_event เมื่อทำการกดปุ่มปิดหน้าต่าง
           โดยจะเรียกใช้ฟังก์ชัน gtk_main_quit() */
```

```
    /* create button */
    button = gtk_button_new_with_lebel("Hello world");
    g_signal_connect(G_OBJECT(button) , "clicked",
                        G_CALLBACK (hello) , "Hello Button pressed");
    /* เป็นการดักเหตุการณ์ที่ชื่อ clicked เมื่อมีการกดปุ่ม Hello world
       โดยจะติดต่อกับฟังก์ชัน hello */

    gtk_widget_show(button);
    gtk_widget_show(window);

    gtk_main();

    return(0);
}
```

จากโปรแกรม Hello2.c จะมีการดักเหตุการณ์อยู่สองที่ดังนี้

```
g_signal_connect(G_OBJECT(window) , "delete_event" ,
                    G_CALLBACK (gtk_main_quit) , NULL);
```

ส่วนนี้จะเป็นการดังเหตุการณ์ delete_event ที่จะเกิดขึ้นเมื่อกดปุ่มปิดหน้าต่าง การดักเหตุการณ์นี้จะไปเรียกใช้ฟังก์ชัน gtk_main_quit() มีผลทำให้หน้าต่างของโปรแกรมถูกทำลาย

```
g_signal_connect(G_OBJECT(button) , "clicked",
                    G_CALLBACK (hello) , "Hello Button pressed");
```

ส่วนนี้จะเป็นการดักเหตุการณ์ที่ชื่อว่า clicked ที่เกิดจากการกดปุ่ม Hello world โดยเมื่อมีการกดปุ่มแล้วจะทำการเรียกใช้ฟังก์ชัน hello พร้อมกับการส่งพารามิเตอร์เป็นสตริงไปเพื่อส่งไปพิมพ์ในฟังก์ชัน hello ต่อไป