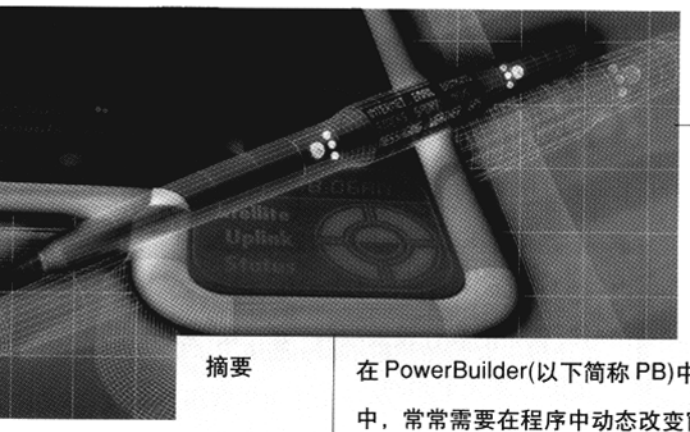
在PowerBuilder中利用参数动态调用图形类型



杨新文 （黑龙江省大庆石化总厂计算机开发公司）

摘要 在PowerBuilder(以下简称PB)中，图形类型是在定义图形的接口中选择的。实际开发应用程序中，常常需要在程序中动态改变窗口中图形的类型。为了提供友好的GUI(图形用户界面)，方便用户使用，使不同的数据以不同的类型看得更清楚，本文介绍了一种利用参数，以图形化的方式动态调用图形类型的方法。

关键词 PowerBuilder 继承 用户对象 用户事件

1 创建图形类型的用户对象

PB提供了各种各样的图形类型，这些图形类型可划分为三个主要的组。第一组包括区图、条图、列图和线图。这些类型的图形的属性是相同的，它们只在显示方式上略有不同。第二组由饼图构成。饼图以百分比方式显示数据。最后一组图形是散射图，散射图没有类型轴。这种图形通常用来比较数据的两个数值集。前两组图形能以二维或三维方式显示在三维方式下，系列号使用了额外的一维，即系列轴，而不再出现于类型轴。我们把上述种类的图形放在一个位图文件中，通过用户对象来选择图形。

打开用户对象画笔，并创建一个新的定制可视用户对象r_graph。为这个用户对象声明一个字符型实例变量is_GraphType，两个整型实例变量ii_Row=0,ii_Col=0。字符型实例变量的作用是根据整型实例变量返回的当前行、当前列的位置确定用户单击的图形种类。用户对象u_graph需要一个函数uf_query_graph()来返回用户单击的当前行、当前列的位置。

函数uf_query_graph()脚本如下：

```
ai_Row=ii_Row
ai_Col=ii_Col
If ii_Row=2 And ii_Col=6 Then
Return 0
ElseIf ii_Row<1 Or ii_Col <1 Then
Return 0
Else
ss_Type=is_GraphType
Return 1
End If
```

表1 函数uf_query_graph()的说明

函数说明		
函数	名称	uf_query_graph
	访问权限	public
	返回值	integer
参数	ai_row	integer
	ai_col	integer
	as_type	string

按照表2所示设置用户对象的属性，以及添加控件。

表2 用户对象u_graph和控件的属性

对象/控件	属性	值
Picture	Name	P_graph
	File Name	graph.bmp
Command Button	Name	cb_ok
	Text	确定
Command Button	Name	cb_cancel
	Text	取消

2 创建图形类型的响应窗口

启动 Windows 画板，创建一个新的响应窗口 W_graph_type。将用户对象放在窗口中，在用户对象和窗口之间使用函数 Open WithParm()通信。为这个窗口声明一个 Graph 型实便变量 igr_Parm，Datawindow 型实例变量 idw_parm,Object 型实例变量 io_passed。PB 为它的每一个应用程序提供了一个唯一的全局消息对象 Message。这个 Message 对象可用于在窗口之间传递参数。用 Message 对象发送的消息包含在 PowerObjectParm 属性中。当调用 Open WithParm()函数时，这些函数会对放在 Message 对象中的一个单一参数进行传递。接收窗口会拷贝三个 Message 对象属性之一的内容到一个变量中去，以使用传入的消息。

按照表 3 所示设置窗口的属性，以及添加控件。

表 3 w_graph_type 和控件的属性

对象/控件	属性	值
Userobject	Name	uo_1
	Visible	true
	Enabled	true

3 创建带有图形控件的图形分析窗口

打开Windows画板，创建一个新的主窗口w_f_ghwc。在此窗口上放置一个数据窗口控件，一个图形控件。为图形控件打开脚本 Script 画板。使用 Declare/User Events 菜单项，声明一个定制的用户事件叫做 graph_type，把这个事件映射到 PB 事件 pbm_custom01 上去。这个过程使得当用户在图形类型窗口上点击时就会触发 w_f_ghwc 窗口的graph_type用户事件。采用参数传递动态改变数据窗口对象。

窗口的 Open 事件脚本如下：

```
int li_row,li_index
dw_1.dataobject=message.stringparm
// 为数据窗口检索数据
dw_1.SetTransObject(sqlca)
dw_1.Retrieve()
gr_1.SetRedraw(FALSE)
integer series_nbr
series_nbr=gr_1.AddSeries(" 产品编码 ")
// 获取系列号
// 通过循环为图形增加数据
li_row=RowCount (dw_1)
for li_index=1 to li_row
gr_1.adddata(1,GetItemnumber(dw_1,li_index,3),
dw_1.getitemstring(li_index,1))
next
gr_1.SetRedraw(TRUE)
```

窗口的 Graph_type 事件脚本如下：OpenWithParm (W_graph_type,gr_1)，这个函数的作用是：打开响应窗口设置图形类型，把图形对象传递给响应窗口。

图形控件不依赖于任何关联的数据集，因此，它必须通过脚本向图形控件输出数据。有许多 PB 函数可用于处理图形内的数据。最先需要用到的函数是那些能生成带有数据的图形函数。AddSeries()函数用来向图形添加新的系列。函数为系列分配了序列号，以便于在后面引用。按照表 4 所示设置窗口的属性以及添加控件。

表 4 w_f_ghwc 和控件的属性

对象/控件	属性	值
DataWindow	Name	dw_1
	Visible	true
	Enabled	true
Graph	Name	gr_1
	Graph Title	供货累计完成
	Category Axis Label	产品编码
	Value Axis Label	商品完成
	Value Axis Round	1

4 创建图形类型的通用菜单

本应用程序的菜单，采用 Windows 的应用程序的功能，即用户最初见到的是“基本菜单”(Short Menu)，当用户需要所有的功能时，见到的是“完整菜单”(Full Menu)。创建两个不同的菜单类 m_menu 和 m_menu_cx，分别接到两个不同的窗口类 w_menu 和 w_menu_cx 上。

打开 Menu 菜单画板，创立框架菜单 m_menu。输入“录入”、“查询”、“报表”、“帮助”、“退出”菜单项。将“帮助”和“退出”菜单项的 Shift Over/Down 属性设置为“真”(TRUE)，这两个菜单项在增加新的菜单项时会自动右移，给新增加的菜单项留出空间。为框架菜单 m_menu 建立一个新的框架窗口 w_menu。为了使得这个框架窗口 w_menu 更加通用，每次菜单在运行中被创建时，都为之提供一个到框架窗口的索引，并且始终使用这个值。定义窗口实例变量 iw_frame，这个变量将存放框架窗口的索

(下转第 57 页)

(上接第66页)

引。由于iw_frame被定义为一个私有变量，菜单应提供一个函数对它的值进行初始化。按照表5中的说明为框架菜单m_menu创建一个函数mf_init()，这个函数用到下面的脚本。

```
//把内部的框架索引置为传递来的值
iw_frame=aw_frame
```

表 5　函数mf_init()的说明

函数说明		
函数	名称	mf_init
	访问权限	public
	返回值	(无)
参数	aw_frame	window

通过继承创建m_menu的后代对象，记作m_menu_cx，增加“图形类型”、“图形分析”菜单项。为菜单m_menu_cx建立一个新的框架窗口w_menu_cx，对w_menu_cx的Open事件编写如下脚本：

```
//取得对窗口的菜单的索引
m_menu ls_menu
ls_menu=this.menuid
//初始化，使知道使用的框架窗口
ls_menu.mf_init(w_menu)
```

“图形类型”菜单项脚本如下：

```
parentwindow.getactivesheet().triggerevent
("graph_type")
```

当用户在带有图形控件的窗口上，点击“图形类型”菜单项时，触发此窗口的图形类型事件，动态改变图形类型。

以上应用在PowerBuilder 6.0下运行通过。■