

第6章 菜单对象

教学提示：上一章学习了窗口对象，本章讲述菜单对象，主要内容包括：创建菜单对象、菜单与窗口的关联、为菜单事件编写脚本、弹出式菜单、菜单的继承等相关知识。重点讲述菜单对象的创建、菜单与窗口的关联、为菜单事件编写脚本、菜单的继承等有关内容。

教学要求：掌握创建菜单对象的方法、菜单与窗口的关联、为菜单事件编写脚本、菜单的继承等知识。上机操作练习，巩固以上所学内容。菜单是在窗口中与用户直接接触的一个重要对象。菜单必须依附于窗口，除了子窗口和响应窗口外，其他类型的窗口都可以带有菜单。PowerBuilder 9.0 为用户提供的菜单有下拉式菜单、级联式菜单和弹出式菜单，另外还有工具图标组合式菜单。PowerBuilder 9.0 为用户提供了灵活方便的菜单设计功能，用户根据自己开发程序的需求和特点可以设计不同风格的菜单。一个清晰、美观的菜单，可以使程序的功能及相关的操作变得更加直观，可以使用户更加方便地操作、运行程序。

6.1 创建菜单对象

本节开始介绍在 PowerBuilder 中如何创建菜单对象。

6.1.1 打开菜单画板

进入 PowerBuilder 后，在系统树窗口中打开工作空间，选择【File】|【New】菜单，或单击工具条中上的【New】图标，弹出新建对象对话框，选择【PB Object】页，如图 6.1 所示。

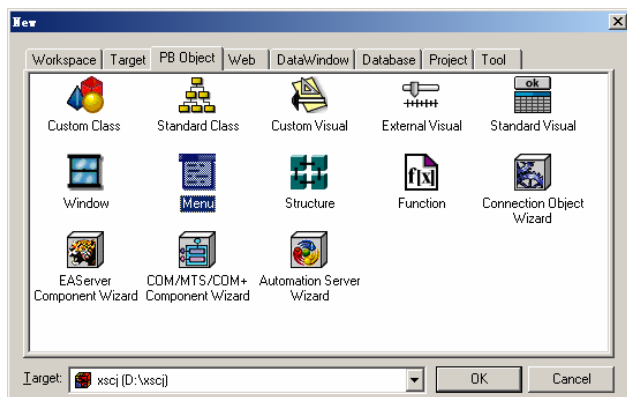


图 6.1 新建对象对话框【PB Object】页

选中【Menu】图标，双击鼠标左键或单击【OK】按钮，即新建了一个菜单对象，默

认名称为：Untitled 0，同时打开了菜单对象画板，如图 6.2 所示。

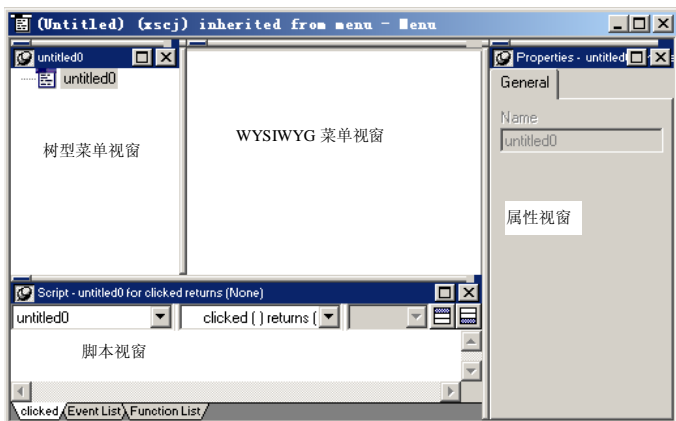


图 6.2 菜单对象画板

用户在 PowerBuilder 9.0 中对菜单的所有操作都是在菜单画板上完成的，因此首先应该认识一下菜单画板的构成。

1. 菜单画板的视窗

菜单画板在缺省情况下打开 4 个视窗，分别为：WYSIWYG 菜单视图、树型菜单视图、脚本视图和属性视窗 4 个子窗口。

WYSIWYG 菜单视图：“WYSIWYG”的含义是“所见即所得(What You See Is What You Get)”，是一个类似于预览形式的菜单显示。可以将用户对菜单所做的所有改动及时地显示出来，所显示的即为应用程序运行时所见菜单形式。

树型菜单视图以树型列表的方式显示当前菜单结构。用户在对菜单进行添加、修改或删除的时候，可以根据自己的需要在这两个菜单视图的任意一个当中进行操作，在改动一个视图中的内容的同时，另外一个视图中的内容也随之发生相应的变化。

脚本视图用于为菜单项编写事件脚本。

属性设置视窗可以查看和设置当前菜单项的各种属性值。

除以上 4 个缺省视窗外，菜单画板布局中还提供了另外两个视窗：Event List 视窗和 Function List 视窗。Event List 视窗用于对当前菜单项脚本中的事件进行列表显示。Function List 视窗是对当前菜单项脚本代码中所用到的函数进行列表显示。

当用户所编写的脚本代码较长时，可以利用这两个视窗迅速地将所要选取的事件或函数进行定位。

2. 菜单工具条

进入菜单操作窗口后，在 PowerBuilder 9.0 主窗口中会出现一个新的工具条，称之为菜单画板工具条，如图 6.3 所示。当然，此时 PowerBuilder 9.0 主窗口中的菜单也会随之变化。



图 6.3 菜单工具条

菜单工具条中从左至右依次为：【Save】(保存)、【Cut】(剪切)、【Copy】(复制)、【Paste】(粘贴)、【Undo】(撤销)、【Redo】(恢复)、【Insert Menu Item】(插入菜单项)、【Delete】(删除)、【Script】(脚本)、【Properties】(属性)和【Close】(关闭)。

菜单工具条中各按钮的功能作用如下。

- 【Save】(保存)：将在当前菜单画板中所作的操作保存。
- 【Cut】(剪切)：将所选中的对象剪切到剪贴板。
- 【Copy】(复制)：将所选中的对象复制到剪贴板。
- 【Paste】(粘贴)：将剪贴板中的对象粘贴到当前指定位置。
- 【Undo】(撤销)：撤销上一步的操作。
- 【Redo】(恢复)：恢复上一步的操作。
- 【Insert Menu Item】(插入菜单项)：从当前指定位置插入菜单项。
- 【Delete】(删除)：删除当前指定对象。
- 【Script】(脚本)：将当前焦点转入脚本视窗。
- 【Properties】(属性)：显示当前指定对象的属性视窗。
- 【Close】(关闭)：关闭当前菜单画板，并弹出提示保存对话框。

6.1.2 创建菜单项

打开菜单对象画板后，可以为菜单对象创建菜单项。

1. 添加菜单栏的菜单项

当要添加菜单栏中显示的菜单项时，操作方法如下。

(1) 在菜单树型视窗中的“Untitled0”上单击鼠标右键，弹出菜单项的快捷菜单，如图 6.4 所示。快捷菜单的基本功能见表 6-1 所示。

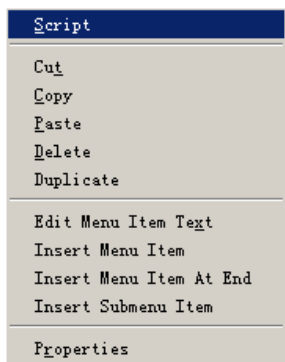


图 6.4 菜单项的快捷菜单

表 6-1 快捷菜单的基本功能

序号	菜单项	功能
1	Script	打开脚本视窗
2	Cut	剪切菜单项
3	Copy	拷贝菜单项

续表

序号	菜单项	功能
4	Paste	粘贴菜单项
5	Delete	删除菜单项
6	Duplicate	复制菜单项
7	Edit Menu Item Text	编辑当前菜单项的显示文本
8	Insert Menu Item	在当前菜单项前插入一个同级菜单项
9	Insert Menu Item At End	在当前菜单的最后插入一个同级菜单项
10	Insert Submenu Item	在当前菜单项下插入一个下级菜单项
11	Properties	打开属性窗口

(2) 选择快捷菜单的【Insert Submenu Item】命令，在“Untitled0”下插入一个子菜单项，显示文本为空，如图 6.4(a)所示，此时可以输入菜单项文本(如输入“系统”)。如果输入菜单项文本后按【Tab】键，则在此菜单项下面自动添加了新的菜单项，如图 6.4(b)所示，这样可以连续添加菜单栏中的多个菜单项。如果输入菜单项文本后按回车键，则结束添加菜单项操作。如图 6.5(c)所示。

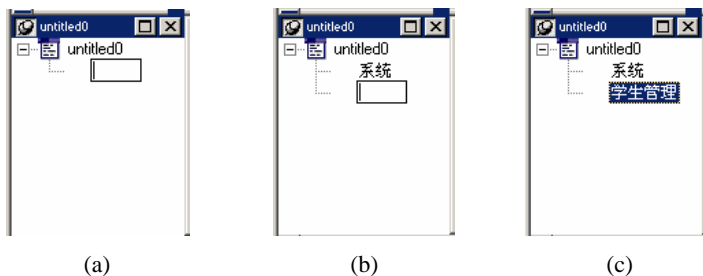


图 6.5 添加菜单栏的菜单项

(3) 如果需要在【学生管理】菜单项下面再添加一个同级菜单项——“课程管理”，则在“Untitled0”的快捷菜单中选择【Insert Submenu Item】命令，也可以在【系统】或【学生管理】菜单项的快捷菜单中选择【Insert Menu Item At End】命令，然后输入菜单项文本“课程管理”。

(4) 如果要在【学生管理】菜单项上面插入同级菜单项——“班级管理”，则在【学生管理】菜单项的快捷菜单中选择【Insert Menu Item】命令，然后输入菜单项文本“班级管理”。

这样，就可以创建出菜单栏上的所有菜单项。

2. 添加下拉式菜单的菜单项

如果要为某个下拉式菜单添加菜单项，比如为【学生管理】菜单添加菜单项，操作方法如下。

(1) 在【学生管理】菜单项上单击右键，在弹出的快捷菜单中选择【Insert Submenu Item】菜单命令，则在【学生管理】菜单下一级生成一个空白的菜单项，如图 6.6(a)所示。然后输入菜单项文本，如“输入学生信息”。

(2) 在【学生管理】菜单的快捷菜单中选择【Insert Submenu Item】命令，或者在【输入学生信息】的快捷菜单中选择【Insert Menu Item At End】命令，可以继续添加【学生管

理】的其他子菜单项，如图 6.6(b)所示。

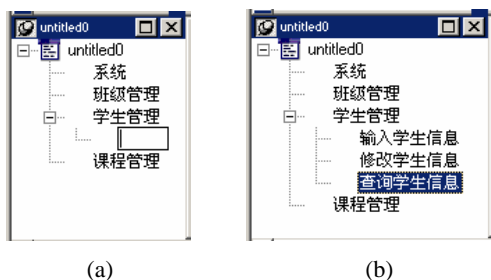


图 6.6 添加下拉式菜单的菜单项

3. 插入级联式菜单项

如果需要为下拉式菜单的菜单项插入级联式菜单项，例如，需要为【学生管理】的【查询学生信息】菜单项添加级联式菜单项，那么选中【学生管理】菜单命令的【查询学生信息】菜单项，单击鼠标右键，弹出其快捷菜单，选择【Insert Submenu Item】菜单命令，在显示的空白文本框中输入级联式菜单项文本，比如“按班级查询”，并按【Enter】键。

依此方法可以添加该菜单的其他级联式菜单项，比如“按学号查询”，“按姓名查询”，如图 6.7 所示。

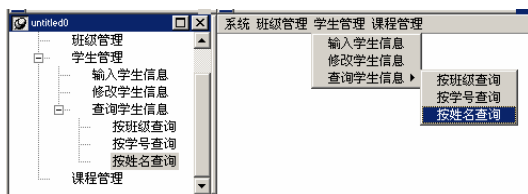


图 6.7 插入级联式菜单项

4. 保存菜单对象

创建完菜单项后，就要保存菜单对象。单击菜单画板工具条上的【Save】按钮，则弹出【Save Menu】对话框，如图 6.8 所示。输入菜单对象的名称，如“m_main”，单击【OK】按钮。

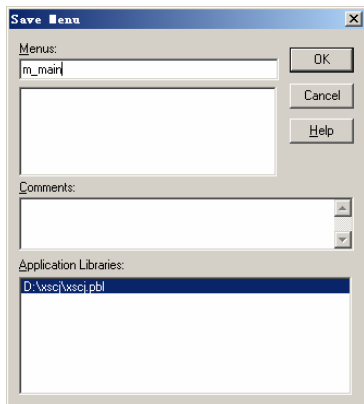


图 6.8 【Save Menu】对话框

如果没有保存菜单对象就关闭菜单对象画板, 那么系统会提示用户保存对象。

6.1.3 编辑菜单

在系统树窗口中双击某个菜单对象, 或者在其弹出的快捷菜单中选择【Edit】项, 即可打开该菜单对象画板, 在菜单对象画板中可以对此菜单对象进行编辑。

1. 插入分隔条

如果要在菜单项之间插入一个分隔条, 首先, 在要插入分隔条的地方插入一个空白菜单项, 然后, 在空白菜单项的文本框中输入一个减号“-”, 按下回车键, 在系统弹出的对话框中, 单击【OK】按钮, 即可完成插入一个分隔条的操作, 如图 6.9 所示。

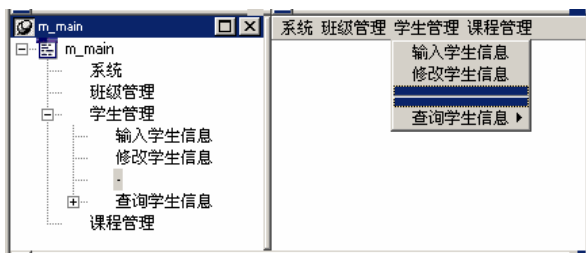


图 6.9 插入分隔条

2. 移动菜单项

菜单对象中的所有菜单项都可以通过鼠标拖曳的方式在当前菜单对象内部进行移动。在菜单画板的 WYSIWYG 视窗或树型视窗中, 选中所要移动的菜单项, 按住鼠标左键拖动, 然后将该菜单项移到新位置后释放鼠标左键, 可以看到原来位置的菜单项已经被移到了新的位置。

3. 删除菜单项

首先, 选中需要删除的菜单项, 单击鼠标右键, 从弹出的快捷菜单中选择【Delete】命令, 既可删除该菜单项。

4. 修改菜单项文本

选中需要修改的菜单项, 单击鼠标右键, 从弹出的快捷菜单中选择【Edit Menu Item Text】命令, 即可修改该菜单项的文本。

5. 添加菜单项的加速键

我们经常看到一些菜单项文本的某个字母带有下划线, 该字母称为此菜单项的加速键 (Accelerator Key), 菜单打开时直接按 Alt+<加速键>即选择了该菜单项。在菜单画板中, 为菜单项添加加速键很简单, 只需在输入菜单项文本时在加速键字母前加上“&”符号即可。

例如, 【系统】菜单项的文本改为“系统(&S)”, 【班级管理】菜单项的文本改为“班级管理(&B)”。如图 6.10 所示为带加速键的菜单。

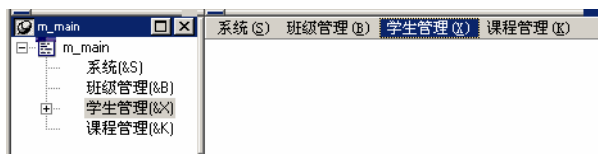


图 6.10 带加速键的菜单

6.1.4 设置菜单项属性

添加一个菜单项后，就会在菜单画板的属性视窗中显示该菜单项的属性，同时也可以设置其属性。菜单项属性视窗包含【General】标签页和【Toolbar】标签页。

1. 菜单项属性的【General】标签页

菜单项属性视窗的【General】标签页用于定义菜单项的基本信息，如图 6.11 所示。

菜单项属性中的【General】标签页中属性的含义解释如下。

- **Name:** 字符串型，指定当前菜单项名称。默认为菜单项文本前加“m_”。
- **Lock Name:** 布尔型，指定当前菜单项是否为锁定状态，在锁定状态下的菜单项不能被修改名称，即修改菜单文本时菜单项名称不随其改变。
- **Text:** 字符串型，指定当前菜单项显示文本，使用中、英文均可。
- **MicroHelp:** 字符串型，微帮助，用来设置当前菜单项在 MDI 界面应用窗口底部的状态栏的提示信息，它比应用程序的联机帮助文档更加直观。
- **Tag:** 字符串型，用来存储与该菜单对象有关的字符串。Tag 属性本身并不做任何事情，其用途比较灵活。
- **Visible:** 布尔型，指定该菜单项是否可视。有效取值为：True，显示；False，不显示。
- **Enabled:** 布尔型，指定当前菜单项是否可用。有效取值为：True，该菜单可用，以正常颜色显示；False，该菜单不可用，以灰色显示。
- **Checked:** 布尔型，指定该菜单项是否已经被选中。有效取值为：True，已经被选中；False，未被选中。
- **Default:** 布尔型，指定该菜单项是否为默认选项。有效取值为：True，默认选项；False，非默认选项。
- **ShiftToRight:** 布尔型，指定当用户在其“子孙”菜单中添加菜单项时，是否可以插入到从该菜单继承得到的菜单项之前。有效取值为：True，可以随意插入菜单项；False，不可以，新菜单项只能添加到其“祖先”菜单项的末尾。

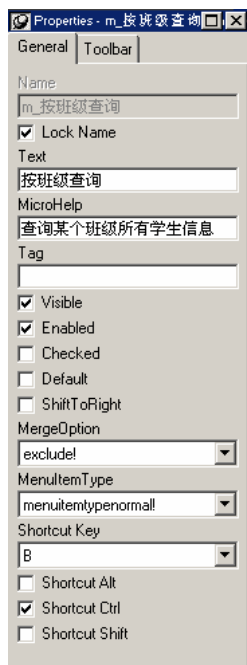


图 6.11 菜单项属性的【General】标签页

- **MergeOption:** Menu Merge Option 类型(枚举类型), 指定当一个 OLE 对象被激活时该菜单的修改方式。有效取值为: EditMenu!、Exclude!、FileMenu!、HelpMenu!、Merge!、WindowMenu!。
- **Menu Item Type:** Menu Item Type 类型(枚举类型), 用来指定所创建的菜单项的类型。有效取值为: MenuItemTypeAbout! (关于类型)、MenuItemTypeExit!(退出类型)、MenuItemTypeHelp!(帮助类型)、MenuItemTypeNormal(一般类型)。
- **Shortcut Key:** 字符串型, 为该菜单项定义快捷键。使用方法为: 在该属性的下拉列表框中选择一个字母键, 再选中复选框【Shortcut Alt】、【Shortcut Ctrl】或【Shortcut Shift】, 就可以与【Shortcut Key】键组合形成快捷键。

例如图 6.11 所示为【按班级查询】菜单项的【General】属性, 菜单项名称为“m_按班级查询”, 菜单项名称处于锁定状态, 微软帮助为“查询某个班级所有学生信息”, 快捷键为【Ctrl+B】。

2. 菜单项属性的【Toolbar】标签页

菜单项属性视窗的【Toolbar】标签页用于定义菜单项的工具条信息, 如图 6.12 所示。当菜单与 MDI 窗口关联时, 菜单的工具条才会显示。

菜单项属性图标【Toolbar】标签页中属性的含义解释如下。

- **ToolbarItemText:** 字符串型, 指定应用程序中与该菜单项相联系的工具条按钮的显示文本信息及提示信息。
- **ToolbarItemName:** 字符串型, 指定当应用程序中与该菜单项相联系的工具条按钮未被按下时所显示的图标名称(系统图标或图标文件名)。
- **ToolbarItemDownName:** 字符串型, 指定当应用程序中与该菜单项相联系的工具条按钮被按下时所显示的图标名称(系统图标或图标文件名)。
- **ToolbarItemVisible:** 布尔型, 指定应用程序中与该菜单项相联系的工具条上的按钮是否可见。有效取值为: True, 可见; False, 不可见。
- **ToolbarItemDown:** 布尔型, 指定应用程序中与该菜单项相联系的工具条按钮的工作状态。有效取值为: True, 默认显示已被按下的状态; False, 显示为未被按下的状态。
- **ToolbarItemSpace:** 整型, 指定应用程序中与该菜单项相联系的工具条上的按钮前需要预留出来的空格数量。缺省值为: “0”。
- **ToolbarItemOrder:** 整型, 指定应用程序中与该菜单项相联系的工具条上的按钮在该工具条上的位置。缺省值为: “0”, 表示按照当前所显示的位置排列。当用户设置为其他非“0”数字后, 则按钮在工具条上的顺序将根据该属性值从小到大排列。

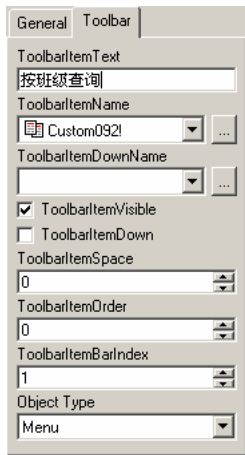



图 6.12 菜单项属性的【Toolbar】标签页

- **ToolBarBarItemIndex**: 整型, 指定应用程序中与该菜单项相联系的图标所在工具条的索引值, 因为菜单项对应的工具条可以有多个。
- **ObjectType**: 枚举类型, 指定工具条上按钮的类型。有效取值为: “Menu”(正常工具条按钮)和“MenuCascade”(级联工具条按钮)。当选择“MenuCascade”类型时, 当前【ToolBar】标签页上会出现“Columns”和“DropDown”两种新的属性。
- **Columns**: 整型, 指定在级联工具条中所显示的列数。
- **DropDown**: 布尔型, 指定该按钮是否以下拉列表框的形式显示在工具条上。有效取值为: True, 以下拉式列表框形式显示; False, 以默认形式显示。

例如, 图 6.12 所示为【按班级查询】菜单项的【ToolBar】页属性, 工具条图标为 , 提示文本为“按班级查询”, 图标在 1 号工具条上。

6.2 为菜单事件编写脚本

6.2.1 菜单事件

菜单编程也是事件驱动型。表 6-2 所示为菜单项的常用事件。

表 6-2 菜单项的事件及含义

事件	含义
Clicked	单击
Help	帮助
Selected	光标移到菜单项

关于菜单事件的说明如下。

1. Clicked 事件

Clicked 事件, 是最常用的事件, 一般应用程序中均通过单击菜单项来触发程序运行。

只有当菜单对象的 Visible 属性和 Enabled 属性的值都设为 True 时, 该菜单对象才会响应鼠标单击或键盘发出的 Clicked 事件。

可触发该事件的情况如下。

- (1) 当鼠标单击菜单项时。
- (2) 当使用键盘选中菜单项(呈现高亮度选中状态), 并按下回车键时。
- (3) 当用户按下该菜单项所对应的快捷键时。
- (4) 当包含该菜单项的下拉式菜单被打开, 并按下该菜单项的加速键时。
- (5) 当弹出式菜单显示时。

2. Help 事件

Help 事件, 用来调用联机帮助文件。

可触发该事件的情况如下。

- (1) 当用户按下键盘上的【F1】按键时。

(2) 从当前窗口的标题栏上拖曳上下文帮助(小问号)按钮到菜单项上时触发该事件。

6.2.2 菜单函数

在编程时常用到关于菜单的函数，下面给出菜单的主要函数的函数名称、返回值数据类型和函数描述。

1. Check

语法: Integer *MenuItemName*.Check()

说明: 将当前菜单项设置为选中状态。如果当前菜单项被选中, 则返回值为“1”, 否则返回值为“-1”; 如果 *MenuItemName* 是 NULL, 则返回值为 NULL。

2. ClassName

语法: String *MenuItemName*.Classname()

说明: 用于取当前菜单项的类名。如果操作成功, 则返回值为该菜单项的类名; 否则, 将返回空字符串。

3. Disable

语法: Integer *MenuItemName*.Disable()

说明: 设置将当前菜单项禁止。如果操作成功, 则返回值为“1”, 否则返回值为“-1”; 如果 *MenuItemName* 是 NULL, 则返回值为 NULL。

4. Enable

语法: Integer *MenuItemName*.Enable()

说明: 设置将当前菜单项激活。如果操作成功, 则返回值为“1”, 否则, 返回值为“-1”; 如果 *MenuItemName* 是 NULL, 则返回值为 NULL。

5. GetParent

语法: PowerObject *MenuItemName*.GetParent()

说明: 用于得到指定菜单对象的“祖先”对象的引用。返回值为“祖先”对象的引用指针。

6. Hide

语法: Integer *MenuItemName*.Hide()

说明: 用于隐藏当前菜单项。如果操作成功, 则返回值为“1”, 否则, 返回值为“-1”; 如果 *MenuItemName* 是 NULL, 则返回值为 NULL。

7. PopMenu

语法: Integer *MenuItemName*.Pop Menu(X_location,Y_location)

说明: 用于在指定位置弹出菜单。如果操作成功, 则返回值为“1”, 否则, 返回值为“-1”; 如果任意一个参数是 NULL, 则返回值为 NULL。

8. Show

语法: Integer MenuItemName.Show()

说明: 用于显示当前菜单对象。如果操作成功, 则返回值为“1”, 否则, 返回值为“-1”; 如果 MenuItemName 是 NULL, 则返回值为 NULL。

6.2.3 为菜单事件编写脚本

为菜单项编写事件脚本的步骤如下。

(1) 在菜单画板的树型菜单视窗或 WYSIWYG 菜单视窗中, 双击要编写事件脚本的菜单项, 则脚本视窗中进入该菜单项的 Clicked 事件, 也可以在事件下拉列表框中选择其他需要编写脚本的事件。

(2) 在脚本视窗中输入脚本代码, 输入完成后单击工具条的【Save】按钮保存菜单对象。

一般应用程序中, 均是通过单击相对应的菜单项来打开与其对应的窗口。下面介绍几个典型的菜单项编程:

1. 打开与菜单相对应的窗口

打开窗口常用到 Open()函数和 Open Sheet()函数。

Open()函数的一般格式为:

Open(窗口名)

例如, 应用程序运行时, 选择【系统】|【修改密码】命令, 利用 Open()函数打开【修改密码】窗口, 如图 6.13 所示。

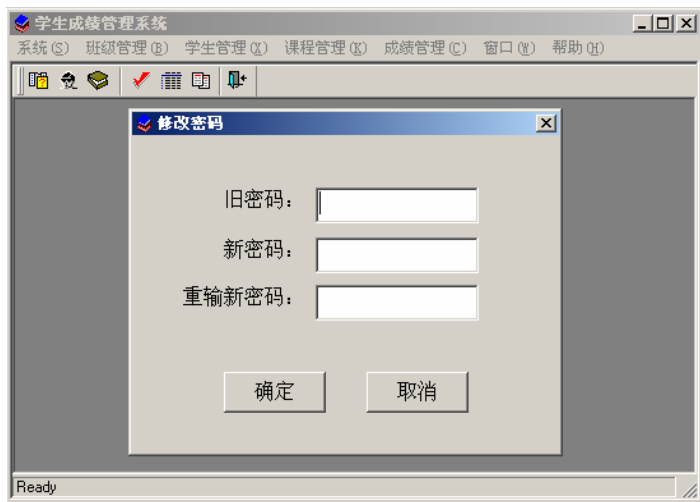


图 6.13 利用 Open()函数打开【修改密码】窗口

从图 6.13 中可知, 利用 Open()函数打开的窗口与父窗口之间是独立的, 在位置上没有制约关系, 窗口以其原始大小、位置显示。其脚本如图 6.14 所示。

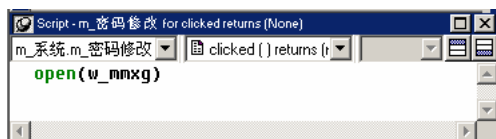


图 6.14 【修改密码】菜单项脚本

OpenSheet()函数的一般格式为:

OpenSheet(<打开的非MDI类型窗口名>,<MDI类型父窗口名>,<与父窗口菜单的相对位置>,<打开方式>)

其中,共有以下3种打开方式。

- (1) Cascaded! : 级联方式,新打开的窗口在父窗口下以级联方式显示。
- (2) Layered! : 层铺方式,新打开的窗口在父窗口下以层铺方式显示。
- (3) Original! : 新打开的窗口以原始大小显示在父窗口下面。

例如,利用 OpenSheet()函数以 Original!方式打开【录入学生基本信息】窗口,如图 6.15 所示,【录入学生基本信息】窗口必须为主窗口。其脚本为:

```
opensheet(w_jiben_shuru,w_main,6,Original!)
```

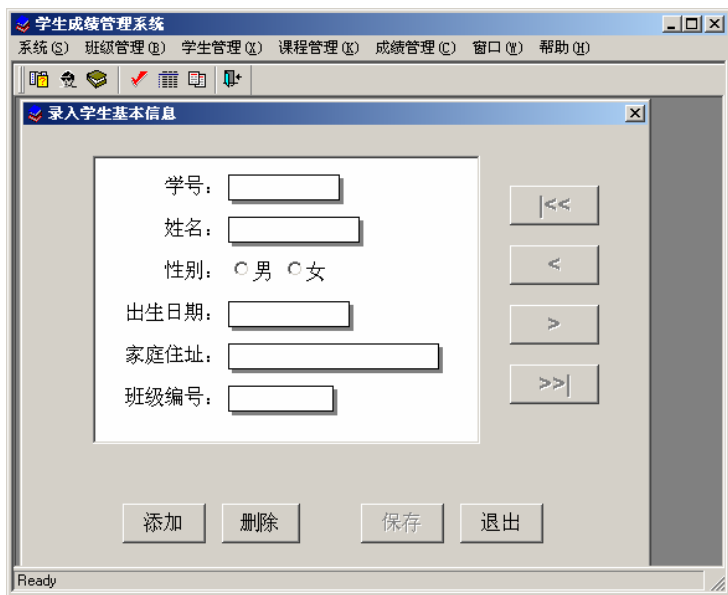


图 6.15 利用 OpenSheet()函数以 Original!方式打开【录入学生基本信息】主窗口

2. 关闭父窗口

一般应用程序中,均是通过单击退出菜单项关闭菜单依附的窗口。这时,要用到 close() 函数。

Close()函数的一般格式如下:

```
close(parentwindow)
```

其中, parentwindow 是指菜单关联的窗口,它具有通用性,不用特别注明窗口名。

3. 用户对菜单项的访问权限

一般应用程序，均是通过授权来控制用户的访问权限。因此，在用户登录成功后，程序要根据用户的访问权限，自动控制主窗口中菜单项的可用与否或可见与否。这时，要利用菜单项的属性 `enabled` 或 `visible`。

如果该菜单项的 `enabled` 属性值为 `True`，则该菜单项正常显示，用户可使用；如果该菜单项的属性值为 `False`，则该菜单项变成灰色或不可见，用户不能使用它。

例如，一般用户没有【系统】|【用户管理】命令的使用权限，假若运行时不令其使用，则在主窗口的 `Open` 事件的对应脚本中给定其属性值为 `False`，代码如下：

```
m_main.m_系统.m_用户管理.enabled=false
```

程序运行时，打开 `w_main` 窗口，打开【系统】下拉式菜单，如图 6.16 所示。

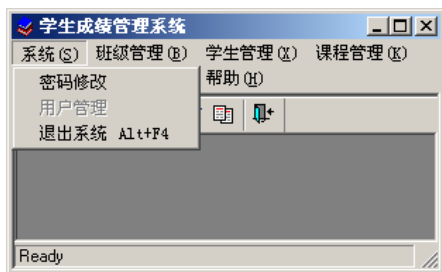


图 6.16 【用户管理】菜单项不可用

由上图可见，主菜单中【系统】|【用户管理】窗口变成了灰色，为不可用状态，用户不能访问【用户管理】窗口。

4. 利用菜单项编程实现窗口按钮的功能

有时为了保持应用窗口界面、风格的一致性，希望均采用菜单操作，此时需要编程将按钮操作变成菜单操作，这时要用到 `TriggerEvent()` 函数。

`TriggerEvent()` 函数的功能是触发指定对象的指定事件。其一般格式为：

```
对象名.TriggerEvent("事件名")
```

例如，在【录入学生基本信息】窗口中有插入、删除、保存和退出 4 个按钮，想通过 `m_op` 菜单实现它们的功能，菜单中包含 4 个对应的菜单项，在对应菜单项的脚本窗口中编写对应的脚本，“`m_插入`”菜单项的脚本如下：

```
w_jiben_shuru.cb_1.triggerevent("clicked")
```

其中，`w_jiben_shuru` 为按钮和菜单依附的窗口名，为了提高脚本的通用性，可以用 `parentwindow` 代替，`cb_ins` 表示“插入”按钮控件名。

说明：此时，在【录入学生基本信息】窗口中必须有 `cb_ins` 按钮，`m_op` 关联到了【录入学生基本信息】窗口，而且该按钮的 `Clicked` 事件已编写了脚本。

6.3 窗口与菜单的关联

菜单必须依附于窗口，除了子窗口和响应窗口外，其他窗口都可以带有菜单。窗口与菜单的关联可以分为静态关联和动态关联两种。

6.3.1 窗口与菜单的静态关联

窗口与菜单的静态关联是常用的关联方法，要实现窗口与菜单的静态关联，操作方法如下：

打开关联菜单的窗口的画板，在窗口属性的【General】标签页的【MenuName】文本框中输入或选择菜单对象名即可，如图 6.17 所示。



图 6.17 为窗口静态关联菜单

菜单必须依附于窗口，但窗口类型不同，菜单的显示结果不同。MDI 类型的窗口中菜单工具条图标可以显示出来，POP 类型和 MAIN 类型的窗口，则不显示菜单工具条图标。

6.3.2 窗口与菜单的动态关联

用户不仅可以在程序设计阶段将窗口与菜单进行静态的关联，还可以通过编写脚本代码，使二者在程序运行的过程中动态关联。

在窗口对象中引用菜单的语法为：

```
Menu.MenuItem    引用菜单项  
Menu.MenuItem.Property    引用菜单项的属性
```

用户有时可能需要在程序运行的过程中，设置或更换当前窗口对象所关联的菜单，这时就用到了函数 `ChangeMenu()`。

例如，当用户希望将当前窗口对象所关联的菜单改为 `m_teacher` 时，只需要在适当的地方加入以下脚本代码即可：

```
w_new.ChangeMenu (m_teacher)
```

其中，`w_new`为关联菜单的窗口。

6.4 弹出式菜单

在应用程序中，一般可以通过单击鼠标右键弹出菜单，称为弹出式菜单。在 PowerBuilder 中，创建快捷菜单有两种：一种是创建关联于窗口的弹出式菜单；另一种是创建无关联的弹出式菜单。

6.4.1 关联于窗口的弹出式菜单

在 PowerBuilder 中，可以将窗口上菜单中某菜单项的下拉式菜单制成弹出式菜单，此时，菜单已挂在窗口上。

下面，以菜单 m_main 为例，说明创建关联于窗口的弹出式菜单的方法。

(1) 将菜单 m_main 挂在窗口 w_main 上，将窗口 w_main 的窗口类型改为“main!”。

(2) 为窗口 w_main 的单击右键(rbuttondown)事件编写脚本，代码如下：

```
m_main.m_学生管理.PopMenu(PointerX(),PointerY())
```

其中，“m_学生管理”为窗口菜单 m_main 的菜单项，且带下拉式菜单；PopMenu()函数的作用是在指定位置显示快捷菜单；PointerX()函数得到鼠标的横坐标；PointerY()函数得到鼠标的纵坐标。因此，PopMenu(PointerX(), PointerY())语句的功能为在鼠标处显示下拉式菜单。

运行时，在窗口 w_main 的任何位置单击鼠标右键，即可弹出快捷菜单 m_学生管理。如图 6.18 所示。

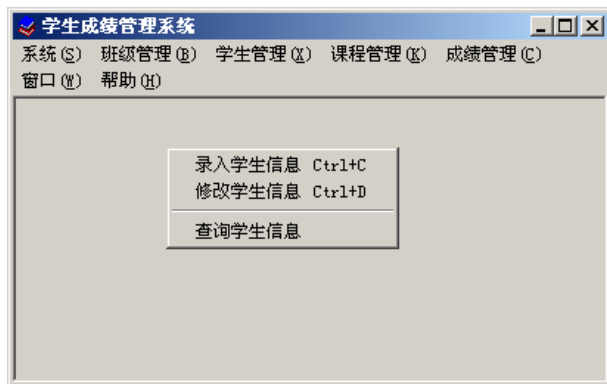


图 6.18 弹出关联菜单的窗口

6.4.2 无关联窗口的弹出式菜单

在应用中，还可将尚未与窗口关联的菜单制作成快捷菜单。

假设 m_op 为创建好的菜单对象，希望在【录入学生基本信息】窗口中单击右键时弹出 m_op 菜单，下面说明创建无关联的弹出式菜单的方法。

(1) 打开窗口 w_jiben_shuru 的画板，使窗口 w_jiben_shuru 成为无关联的菜单项。

(2) 为窗口单击鼠标右键事件编写脚本, 代码如下:

```
m_op m_popup  
m_popup=Create m_op  
m_popup.PopMenu(PointerX(),PointerY())
```

说明: 首先定义菜单对象 `m_op` 的实例变量 `m_popup`, 利用 `Create` 语句创建该实例变量, 利用该实例变量弹出快捷菜单。

(3) 运行时, 在窗口 `w_jiben_shuru` 的任何位置单击鼠标右键, 即可弹出快捷菜单 `m_op`, 如图 6.19 所示。

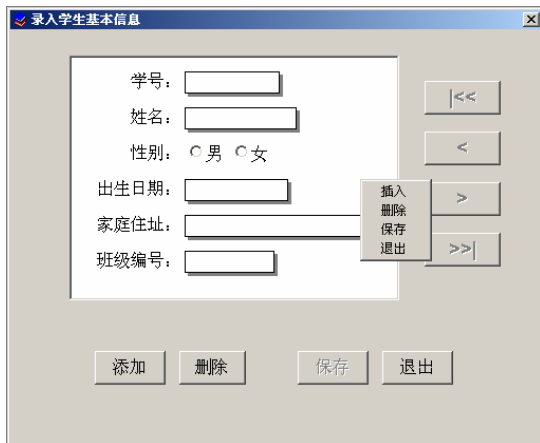


图 6.19 弹出无关联菜单的窗口

6.5 菜单的继承

6.5.1 通过继承创建菜单

PowerBuilder 是面向对象的数据库应用程序开发环境, 支持继承方法。为了提高程序的编写速度, 编程时一般尽可能采用继承方法。

当用户希望新生成的菜单与某一个现有的菜单相似, 并沿袭其风格时, 可以使用继承方式生成新的菜单, 保留其“祖先”菜单的事件、函数、结构、变量以及脚本等信息。用户可以在这些通过继承方式导入的信息的基础上, 对新生成的菜单进行适当的修改与扩展, 以达到新的要求。通过继承方式生成新菜单可以缩短编码时间、提高效率。

通过继承方式, 生成新菜单的操作步骤如下。

(1) 在菜单栏中选择【File】|【Inherit】命令, 或单击主工具条的【Inherit】图标, 打开【Inherit from Object】对话框, 如图 6.20 所示。

(2) 在【Inherit from Object】对话框中, 首先选择所要继承的对象类型“Menu”, 然后在所列出的该类对象名称中, 选择要继承的“祖先”菜单对象, 最后单击【OK】按钮, 系统将会以继承方式创建一个所选菜单对象的派生对象, 同时打开该菜单对象的画板。

(3) 编辑此菜单对象, 在祖先菜单上进行扩张。

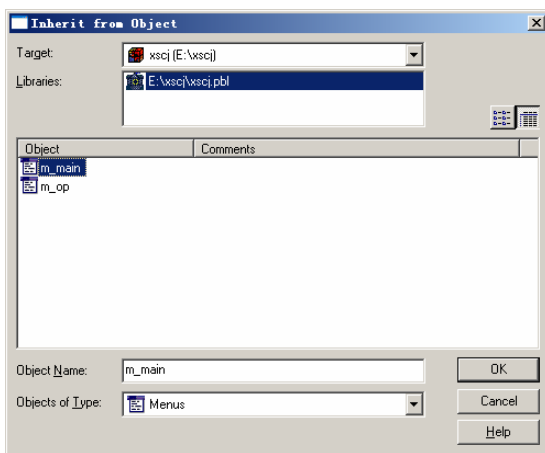


图 6.20 【Inherit from Object】对话框

(4) 在菜单栏中选择【File】|【Save】命令，或单击画板工具条的【Save】图标，系统将显示【Save Menu】对话框，保存利用继承方式新建的菜单。

说明：保存通过继承方式生成的菜单时，必须先关闭其“祖先”菜单；否则，系统将会弹出【Inherit Failed】警告对话框。

6.5.2 继承菜单的修改

对于继承的菜单项，用户可以在继承的基础上做出修改，可以增加菜单项、修改菜单项属性(部分)、增加菜单项的脚本语句等。但是，其“祖先”菜单的程序是只读的，不能修改。

用户可对由继承方式生成的新菜单做出修改。具体如下。

- (1) 在当前菜单的末尾增加新的菜单项。
- (2) 在当前菜单中有限制地插入新的菜单项。
- (3) 改变现有菜单项的 Text 等属性，或使不需要的菜单项变为不可见状态。
- (4) 为在“祖先”菜单中不含有脚本的菜单项增加脚本。
- (5) 对从“祖先”菜单中继承的脚本进行扩展或删除。
- (6) 为新生成的“子孙”菜单对象声明函数、结构和变量。

虽然用户可以从继承生成的新菜单中得到许多的便利条件，但是在对它进行操作时也会有一些限制，具体如下。

- (1) 不能改变各菜单项的顺序。
- (2) 不能删除某一个通过继承方式得到的菜单项。
- (3) 不能在两个由继承得到的不可移动的菜单项之间插入新的菜单项。
- (4) 不能改变由继承得到菜单项的 Name 属性。
- (5) 不能改变由继承得到的菜单项的类型。

6.5.3 查看“祖先”菜单脚本

查看“祖先”菜单脚本的方法是：在脚本窗口中第 3 个下拉列表框中选中“祖先”菜单，则“祖先”菜单的程序就显示出来，如图 6.21 所示。

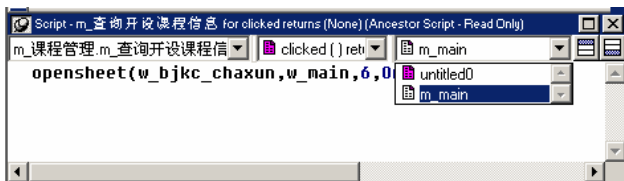


图 6.21 “祖先”菜单程序显示窗口

菜单的“祖先”菜单名称为 m_main，所以在第 3 个下拉列表中选择 m_main，即可显示在 m_main 中的程序。

注意：继承菜单后，在脚本窗口中第 2 个下拉列表框中事件图标为粉红色；如果该事件仅在“祖先”菜单中有程序，则图标为全粉红色；如果该事件不仅在“祖先”菜单中有程序，而且在当前菜单中也有程序，则图标为半粉红色。

6.5.4 查看菜单的继承层次

查看菜单继承层次的方法与窗口相同，单击主工具条中的【Browser】图标，弹出目标浏览窗口，选择【Menu】标签页，如图 6.22 所示。

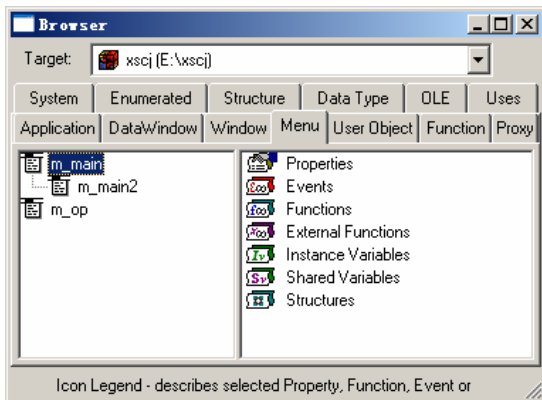


图 6.22 查看菜单继承层次窗口

选中要查看继承层次的菜单右击，在弹出的快捷菜单中选择【Show Hierarchy】命令，该菜单的继承层次就显示出来。本例中，显示菜单 m_main2 是由 m_main 继承而来。

6.6 小 结

本章讲述了创建菜单对象、菜单与窗口的关联、为菜单事件编写脚本、弹出式菜单、菜单的继承等相关知识。重点讲述了菜单对象的创建、菜单与窗口的关联、为菜单事件编写脚本、菜单的继承等知识。

学生应熟练掌握创建不同菜单对象的方法，掌握菜单与窗口的关联、为菜单事件编写脚本、菜单的继承等有关知识。

能上机操作练习，巩固以上所学内容。

6.7 实 训

实训目的

- (1) 掌握菜单的创建、修改、删除操作。
- (2) 掌握菜单的属性设置及引用方法。
- (3) 掌握将菜单挂在窗口上的方法。
- (4) 掌握弹出式菜单的创建方法。
- (5) 掌握利用继承创建菜单的方法。
- (6) 掌握为菜单事件编写脚本。

实训内容

- (1) 创建学生成绩管理系统的菜单。
- (2) 继承方式创建菜单。
- (3) 创建弹出式菜单。

实训步骤

1. 创建学生成绩管理系统的菜单

学生成绩管理系统的菜单如图 6.23 所示。

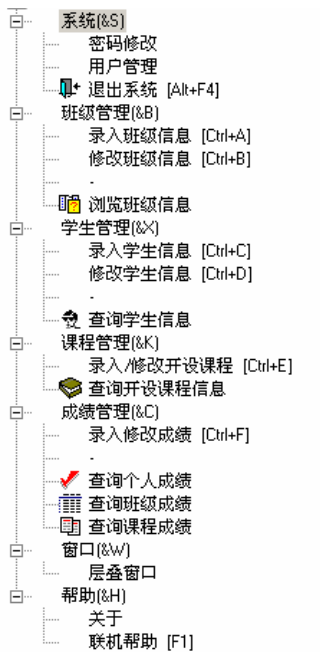


图 6.23 学生成绩管理系统菜单

- (1) 创建工作空间、应用对象、主窗口 w_main，类型为“main!”。并使运行能打开主

窗口。

- (2) 创建主菜单 `m_main`，包含图中的菜单项。
- (3) 设置所有菜单项的属性。
- (4) 将 `m_main` 挂在 `w_main` 上，运行程序。
- (5) 为【系统】菜单项编写脚本，分别利用 `open()` 函数和 `opensheet()` 函数打开对应的窗口(自己创建)，查看运行结果。
- (6) 改变 `m_main` 菜单某一菜单项的 `enabled`、`visible` 和 `checked` 等属性，查看运行结果。
- (7) 修改主窗口 `w_main` 的 `Open` 事件脚本，动态改变 `m_main` 菜单某一菜单项的 `enabled`、`visible` 和 `checked` 等属性，查看运行结果。
- (8) 把主窗口 `w_main` 的类型设置为 `MDIhelp!`，查看运行结果。

2. 继承方式创建新的菜单对象

- (1) 从 `m_main` 继承方式创建新的菜单对象 `m_mainchild`，增加【操作】菜单项，其下包含【添加】、【删除】和【保存】命令。
- (2) 将主窗口关联到 `m_mainchild` 菜单上，查看运行结果。
- (3) 查看各菜单的继承层次。

3. 创建弹出式菜单

- (1) 利用创建关联快捷菜单的方法，在 `w_main` 窗口中创建 `m_main` 菜单的“m_班级管理”菜单项的快捷菜单。
- (2) 利用创建无关联快捷菜单的方法，在 `w_main` 窗口中创建 `m_popmenu` 的快捷菜单，`m_popmenu` 为新建的菜单对象。

6.8 习 题

1. 简述利用“向导”生成菜单的步骤。
2. 简述利用继承方法制作菜单的步骤。
3. 简述将菜单与窗口关联的方法，列出能挂菜单的窗口类型。
4. 简述查看菜单继承层次的方法。
5. 简述查看继承菜单“祖先”脚本的方法。
6. 简述创建快捷菜单的方法。