第 5 章　窗 口 对 象

教学提示：窗口是 PowerBuilder 中的一个重要对象，它是用户与 PowerBuilder 应用程序之间的一个接口，是一种基于 Windows 的友好界面。用户通过窗口与应用程序实现交互。应用程序的主要操作都是在窗口上实现的。构思新颖、设计独到、条理清晰的应用程序的窗口往往会让人赏心悦目，回味无穷。

教学要求：本章主要介绍窗口设计的一些基本方法和技巧。美化窗口界面，使其更具感染力。

5.1　创建窗口对象

窗口是应用程序的图形用户界面的基本元素，其他的应用程序组件往往是通过窗口联系起来的。对于一个应用程序来说，创建窗口对象是极其基本的一步工作，也是极其重要的一步。

5.1.1　创建窗口

PowerBuilder 已经提供了最基本的窗口框架，只要单击几下鼠标就可以得到一个空白的窗口。创建新窗口的步骤如下。

(1) 单击工具栏上的【New】图标按钮，弹出 New 属性页对话框。

(2) 选择【Object】页。

(3) 双击【Windows】图标或选中【Windows】图标后，单击【OK】按钮。

这样，就创建了一个新的空白窗口界面，如图 5.1 所示。窗口对象命名时的默认前缀为“W_”。

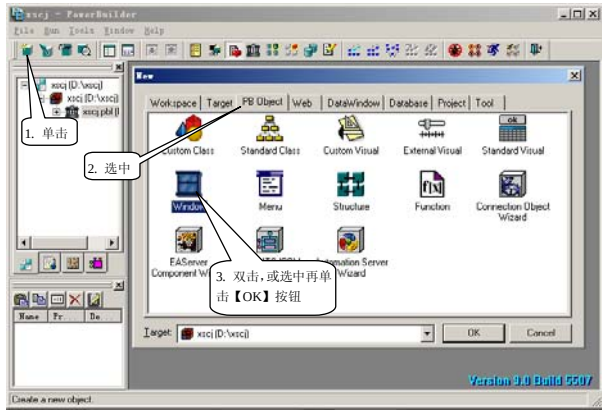

图 5.1　创建新窗口的步骤

5.1.2　窗口画板

窗口画板是由布局视窗、窗口属性区、函数列表区、脚本编辑区、结构列表区、结构定义区、控件列表区、事件列表区以及非可视对象列表区等区域组成的。其中，布局视窗和窗口属性区是最常用的，一般要保持打开状态，其余区域可以根据需要打开或关闭。各区域的大小可以随意调节，各区域在窗口画板中的位置也可以自行设置，方法是将光标移到标题栏处，按下左键拖动到适当的位置即可。区域的打开可以在【View】菜单项下进行选择，如图 5.2 所示。关闭某个区域只要单击区域右上脚的“X”标志即可，关闭整个窗口画板可以使用工具栏上的“Close”图标。

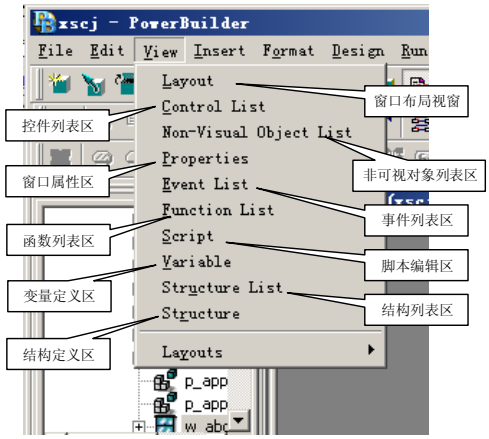

图 5.2　打开窗口对象画板的区域

5.1.3　预览窗口

在窗口的设计过程中，可以随时预览设计窗口在实际运行时的外貌。

1. 预览的方法

(1) 从菜单中选择【Design】|【Preview】命令，建好的窗口即以执行方式显示出来(这就是应用程序运行时该窗口显示的样子)。

(2) 直接使用窗口对象工具条上的【Preview】图标实现预览。

(3) 使用快捷键 Ctrl+Shift+P。

2. 预览窗口可以进行的操作

在预览设计好的窗口时，除了获得对它外观的认识外，还可以进行如下的操作。

(1) 移动窗口。

(2) 改变它的大小(如果窗口的 Resizable 选项是打开的，即窗口大小是可以改变的)。

(3) 最大化和恢复它(如果窗口的 Maximize Box 选项是打开的，即窗口是可以改变大小的)。

(4) 在窗口中跳转输入的焦点(用 Tab 键)。

(5) 选择响应的控件。

3. 在预览窗口时不能进行如下操作

(1) 改变窗口的属性。在预览窗口时做的改变，例如改变它的大小，在退出预览状态时是不能保存下来的。

(2) 触发事件。例如，在预览窗口时单击某个命令按钮不会触发它的 Clicked 事件，即事件中的脚本将不会执行。

(3) 连接数据库。由于预览状态不会执行程序脚本，所以所有与数据库的连接不会被建立。

5.2　设置窗口对象的属性

窗口对象包括属性、函数和事件 3 项特性。属性决定窗口的外观和行为，事件说明了应用程序能够响应用户操作的操作类型，函数为持续控制窗口提供了安全而简洁的手段。

窗口属性不同，呈现给用户的界面就有所区别，可以根据需要来设置窗口的属性。打开要设置属性的窗口，在窗口的任意空白处单击右键，选择【Properties】菜单，或选择【View】|【Properties】命令，就打开了窗口的属性窗口。窗口的属性显示采用标签页面，每一页的作用如表 5-1 所示，其中最重要的是基本特征属性页。下面将分别介绍各属性页的具体内容。

表 5-1　窗口各属性页的作用

窗口属性表页的名称	表页的英文名称	功　　能
基本特征属性页	General	窗口名称、特征、外观、风格等设置
滚动条属性页	Scroll	窗口滚动条及其滚动速度的设置
工具栏属性页	Toolbar	工具栏的位置和几何尺寸的设置
其他属性页	Other	窗口大小和窗口内光标形状的设置

5.2.1　窗口的类型

窗口的类型可以决定不同窗口之间的相互关系，而且不同类型的窗口可以完成不同的操作，对用户交互操作时要求的响应也是不同的。

窗口一共有 6 种类型：

- Main(主窗口)
- Child(子窗口)
- Popup(弹出式窗口)
- Response(响应式窗口)
- MDI(多文档界面)
- MDIHelp(带微帮助的多文档界面窗口)

1. Main(主窗口)

主窗口是可以独立存在、不依赖于任何其他窗口的窗口。在执行应用程序的过程中，

获得焦点时会覆盖其他窗口，失去焦点时又会被其他窗口所覆盖。

它可以被最大化、最小化，也可以嵌入菜单。但菜单中若有工具栏，则工具栏不能被显示。若在显示菜单的同时，也要显示与菜单项对应的工具栏，则应选择 MDI 或 MDIHelp 类型的窗口。

2. Child(子窗口)

子窗口和它的父窗口相关联，因为它一定要放在父窗口的区域内。它的父窗口可以是主窗口或响应式窗口。可以在父窗口中移动子窗口，但是不能移动到父窗口之外。当移动父窗口时，子窗口随着一起移动。

子窗口没有菜单，并且不能被当作活动窗口。它们可以有标题条，可以被最大化、最小化，并且可以改变大小。不过它的改变都只是在父窗口内进行的，也就是说当它最大化时，它将填满其父窗口的空间；当它最小化时，它变成一个图标放在父窗口的最下角。更重要的是，如果父窗口最小化了，那么它的所有子窗口都要最小化；如果父窗口关闭了，它的所有子窗口都要随着一起关闭。

3. Popup(弹出式窗口)

弹出式窗口通常由另一个窗口打开，可以打开它的窗口称为它的父窗口。它可以覆盖父窗口，也可以移出父窗口，但不能被父窗口覆盖。它总是显示在父窗口的前面。

弹出式窗口可以被最大化、最小化，也可以拥有自己的菜单。当弹出式窗口被最小化时，它以图标形式显示在屏幕底部，而不是在父窗口中；当弹出式窗口最大化时，最大化到整个屏幕，而不仅仅覆盖父窗口部分；当父窗口最小化时，弹出式窗口随其隐藏，当父窗口被关闭时，弹出式窗口也随之关闭。

弹出式窗口一般用来显示父窗口中一个字段或者运行一些额外的信息。一般的对话框都是弹出式窗口。

4. Response(响应式窗口)

响应式窗口是一类不太受人欢迎的窗口，因为它要求用户必须首先对这个窗口的消息做出响应，然后才能继续执行应用程序。MessageBox()函数提供的窗口就是响应式窗口。

5. MDI(多文档界面窗口)

MDI 窗口是一个最先打开的窗口，它充当其他窗口的容器。任何时候在 MDI 中打开的窗口叫做一个工作表，工作表只能在框架内活动，若把它极小化，就变成一个位于框架底部的图标。

6. MDIHelp(带微帮助的多文档界面窗口)

MDIHelp 类似于 MDI 窗口，但 MDIHelp 窗口在底部多了一个状态栏，用于向用户显示当前应用程序的一些简短信息和帮助信息。

5.2.2 窗口的主要属性

窗口的基本特征属性页如图 5.3 所示。

1. 设置【General】标签页

(1) 指定窗口类型：单击【WindowType】下拉式列表框，从中选择合适的窗口类型。

(2) 指定窗口菜单：在【Menu Name】后面指定菜单名称。

(3) 选择窗口图标：单击【Icon】属性下拉列表框右边按钮，从中选择一个系统预定义图标，或单击右边的 Browse 按钮指定一个图标文件即可。

(4) 设置窗口颜色。

设置窗口的背景颜色：从【BackColor】下拉列表框中选择颜色。

设置 MDI 窗口工作区颜色：从【MDIClientColor】下拉列表框中选择颜色。

(5) 指定其他基本特征。

① Title：窗口的标题，默认的标题是 Untitled，有的窗口类型可以输入标题，有的不能输入标题，这可根据该项是否是可输入状态而定。

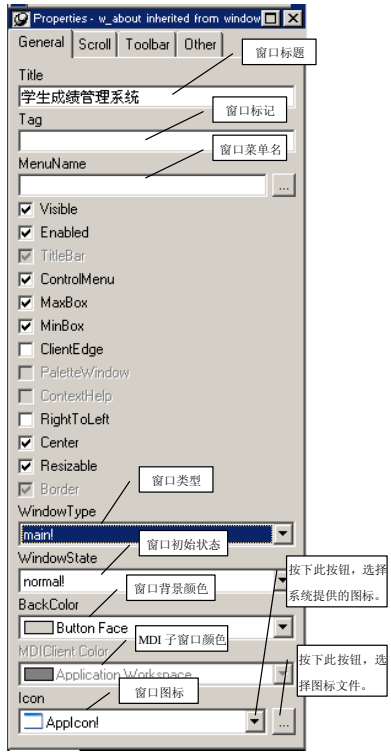

图 5.3　窗口的基本属性页

② Visible：选择该项，表示打开的窗口处于可视状态，否则窗口虽然已经打开但不能显示在屏幕上。在程序运行时通过脚本控制，能根据需要决定是否显示窗口。其语法格式为：

窗口名称.`visible=True` 表示显示窗口；窗口名称.`visible=False` 表示不显示窗口。

③ Enabled：选择该项表示窗口处于可使用状态，否则窗口及窗口中的控件不能使用。该项也可通过脚本来控制，使得程序在运行时能根据需要决定窗口是否可用。其语法

格式为：

窗口名称.enabled =True 表示窗口可用；窗口名称.enabled=False 表示窗口不可用。

④ Control Menu：选择该项表示在窗口的标题栏上可以使用控制框，包括窗口左上角的控制菜单项、窗口右上角的关闭控制项；不选择该项表示不能使用窗口标题栏上的所有控制框。

⑤ MaxBox：在选择 Control Menu 的同时再选择该项，则在窗口的右上角显示最大化按钮。单击最大化按钮，窗口变为最大化状态同时最大化按钮变为还原按钮，单击还原按钮回到窗口的初始状态。

⑥ MinBox：在选择 Control Menu 的同时再选择该项，则在窗口的右上角显示最小化按钮，单击最小化按钮，窗口在屏幕底部缩为一个图标，单击图标回到窗口的初始状态。

⑦ Border：指定窗口周围是否要加边框，只有子窗口、弹出式窗口可以不加边框，其他窗口系统自动加上边框，并且用户不能将边框去除。

⑧ Resizable：指定在程序运行过程中，用户是否可以改变窗口的大小，系统指定响应窗口不能改变其大小。

⑨ WindowState：指定窗口第一次显示在屏幕上的方式。这里方式有 3 种：Normal 表示按照窗口定义的大小和位置显示在屏幕上，Maximized 表示以最大化显示窗口，Minimized 表示以最小化显示窗口。

2. 设置【Scroll】标签页

该页用于定义滚动条参数。

(1) HScroll Bar：指定窗口中是否具有水平滚动条。窗口显示的大小不能无限增大，有时希望看到区域右边内容，可选择该项。

(2) VScroll Bar：指定窗口中是否具有垂直滚动条。当窗口垂直方向内容较多，希望能看到显示区域以外内容，可选择该项。

(3) UnitsPerLine：单击垂直滚动条上下箭头时，垂直滚动条滑块每次上下移动的单位数。当值为 0 时，每次滚动窗口高度的 1/100。

(4) UnitsPerColumn：单击水平滚动条左右箭头时，水平滚动条滑块每次左右移动的单位数。当值为 0 时，每次滚动窗口宽度的 1/100。

(5) ColumnsPerPage：单击水平滚动条上的任意位置，滑块左右滚动的列数。默认情况下该值为 0，滚动 10 列。

(6) LinesPerPage：单击垂直滚动条上的任意位置，滑块上下滚动的行数。默认情况下该值为 0，滚动 10 行。

5.2.3 窗口的工具栏

应用程序窗口的工具栏属性通过【Toolbar】标签页来设置。

注意，这里所说的工具栏并非在 PowerBuilder 编程环境中系统提供的工具栏，而是指应用程序制作的在应用程序中使用的工具栏。

(1) ToolbarVisible：定义窗口是否显示工具栏。

(2) ToolbarAlignment：设置工具栏的位置。5 个选项：alignatbottom、alignatleft、

alignatright、alignattop、floating 分别表示工具栏位于底部、左边、右边、顶部、浮动状态。

若工具栏显示位置为 floating，则应填写工具栏左上角的 X，Y 坐标值，并指定工具栏的宽度和高度。

(3) ToolbarX、ToolbarY：表示工具栏左上角的 X 坐标、Y 坐标，该坐标是相对于窗口的。

(4) ToolbarWide、ToolbarHeight ：表示工具栏的左右长度、上下高度，这是绝对的、不是相对窗口。

5.2.4　窗口的其他属性

(1) X：窗口左上角距屏幕或其父窗口客户区左边界的距离。

(2) Y：窗口左上角距屏幕或其父窗口客户区上边界的距离。

(3) Width：窗口的左右宽度。

(4) Height：窗口的上下高度。

(5) Pointer：鼠标移到该窗口区域内时鼠标的形状，可以单击下拉列表框，选择系统预定义的形状，或单击右边按钮，选择文件来定义鼠标形状。

5.3　窗 口 函 数

前面介绍的新窗口的创建以及窗口属性的配置提供了不需要编写任何程序代码就得到所需窗口的便利，这正是可视化编程的一大特点。但是，要设计出真正使用和动态的应用程序，编写程序代码是必不可少的。在前面的章节中，已经介绍了 PowerBuilder 编写程序的 PowerScript 语言以及常用的标准函数，在这里将介绍 PowerBuilder 系统专门为窗口提供的一些函数以及用户自定义函数。

5.3.1　PowerBuilder 的窗口函数

PowerBuilder 提供了一组函数用于窗口操作，这组函数既包括了系统函数，也包括窗口对象函数。下面介绍几个常用窗口函数及其使用方法。

1. 打开窗口 Open()和关闭窗口 Close()

Open()和 Close()函数是两个特别常用的函数。Open()函数的作用就是用来打开一个窗口，它的参数是要打开的窗口的名字。如打开名叫 w_login 的窗口使用：Open(w_login)。而 Close()函数的作用恰好相反，它是用来关闭一个窗口，它的参数就是要关闭的窗口的名字。如关闭名叫 w_login 的窗口：Close(w_login)。

2. 显示窗口 Show()和隐藏窗口 Hide()

Show()函数的作用是用来显示窗口的，要显示 w_login 窗口使用：w_login.Show()。它与把窗口的 Visible 属性设置为 True 的结果是相同的，即等同于语句：w_login.Visible=True。

Hide()函数的作用是用来隐藏窗口，即要隐藏 w_login 窗口：w_login.Hide()。它与把窗口的 Visible 属性设置为 False 的结果是相同的，即等同于语句：w_login.Visible=False。

3. 移动窗口 Move()

Move()的作用是将窗口移动到指定的位置，它的参数(x，y)是移动目标点的位置。将窗口 w_login 移动到(100，200)处：w_login.Move(100，200)。

4. 改变窗口大小 Resize()

Resize()函数的作用是改变窗口的高度和宽度，它的参数(width，height)是窗口的新的高度和宽度。将窗口 w_login 的尺寸放大 1 倍：w_login.Resize(w_login.width*2, w_login.height*2)。

5.3.2　用户自定义窗口函数

在事件脚本编程中，编程人员可以自定义一些窗口函数。使用自定义函数的优点是程序简洁明了、易于维护，并且代码实现共享，移植方便。下面介绍在 PowerBuilder 中，定义和使用用户自定义函数的步骤。

1. 进入函数区

如果函数定义区没有打开，则可以用下列两种方法之一将其打开。

(1) 选择【Insert】|【Function】命令，如图 5.4 所示。

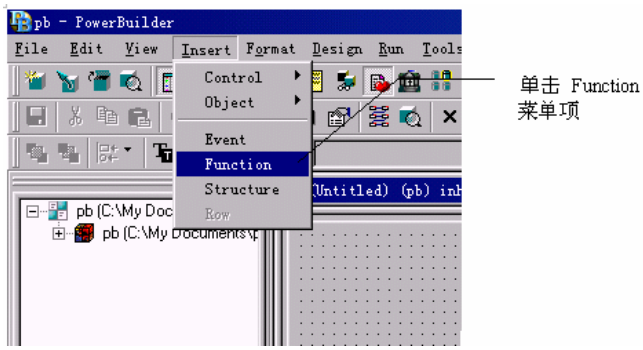

图 5.4　进入函数定义区方法一

(2) 单击脚本子窗口左上边的下拉列表框的小三角，选择弹出列表选项中的【Function】选项，如图 5.5 所示。

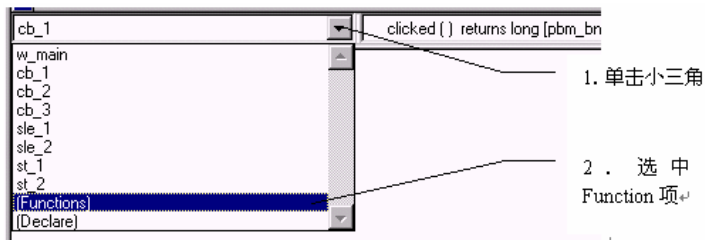

图 5.5　进入函数定义区方法二

2. 函数定义区

弹出的函数定义区如图 5.6 所示。在【Access】列表框中选择函数返回值的访问控制范围，有关的作用域参见说明。在【Return Value】栏中选择返回值的数据类型。在【Function

Name】列表框中输入函数的名称。接着，定义函数的入口参数，入口参数可以没有，也可以有多个。在【Value】下拉列表框中选择入口参数的传递方式，参见表 5-2。在【Argument Type】下拉列表框中选择入口参数的数据类型，在【Argument Name】栏中输入入口参数的名称。

表 5-2　入口参数的传递方式

传递方式	名称	说明
By value	值传递	传递给函数的参数是原变量的一个拷贝，所以在函数内对参数的修改不会影响原变量
By reference	址传递	将原变量的指针传递给函数，所以在函数内对参数的修改也就是修改了原变量
Read-only	只读传递	原变量的值作为只读，即不可修改的量传递给函数

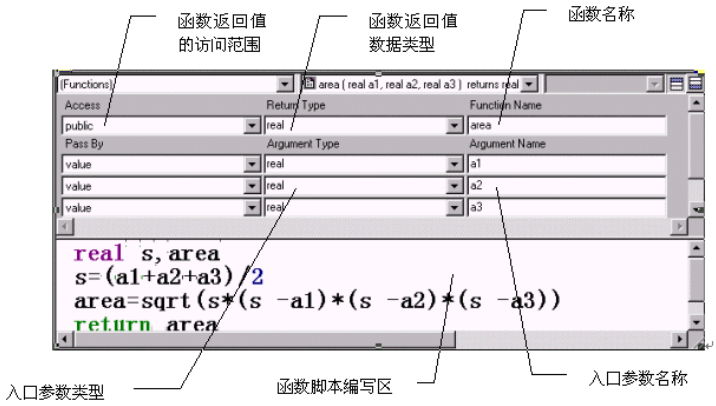

图 5.6　函数定义区

3. 编写函数代码

在函数定义区下部的脚本区域编写函数代码，完成函数的定义。

4. 函数的使用

函数的使用有静态调用和动态调用两种方式，前者是系统默认的函数调用方式，应用最多。所谓静态调用，就是系统在编译代码时就对函数进行彻底的编译，对返回值以及入口参数进行检查和匹配，出现问题立即报告错误。而动态调用的函数在程序执行的时候才会去查找和调用相应的函数，而在程序编译时可以没有该函数。其优点是程序的开发具有极大的灵活性，缺点是降低了应用程序执行的速度，缺少调试编译中的错误检查功能。

5.4　窗 口 事 件

PowerBuilder 程序设计的一个显著的特点是客户程序和函数都是由事件触发的，编程人员需要在某一个事件发生的时候进行相关的处理。窗口事件是捕捉和处理影响整个窗口

的用户和系统动作的地方。窗口也有许多的事件，了解这些事件的发生时机对于程序设计是非常有益的。

5.4.1 Activate 事件

Activate 事件在窗口成为活动窗口时、得到输入焦点之前触发，并且窗口中 Tab 值最小的那个控件得到输入焦点。假如窗口上没有可见的对象，那么窗口本身得到输入焦点。

Activate 事件没有参数，只有唯一的返回值 0，表示继续操作。

Activate 事件常常用来更新工作表管理器，指明哪一张工作表是当前的，或者被用来变换菜单的可用和不可用状态。

5.4.2 Open 事件

打开窗口时触发该事件，它发生在窗口被打开之后，未显示之前。

这个事件没有参数，只有唯一的返回值 0，表示继续操作。

当窗口的 Open 事件被触发时，也会触发一些其他事件，如 Activate 事件和 Show 事件，窗口上各个控件的 Constructor 事件等。

通常在该事件中初始化窗口中的一些变量和控件，也可以设置数据窗口和事物对象的关联关系等。

如在窗口的 Open 事件中将数据窗口控件与事务对象连接并检索数据。

```
dw_1.settransobject(sqlca)
dw_1.retrieve()
```

5.4.3 Clicked 事件

当用户单击窗口中没有被控件覆盖的地方，在释放鼠标左键之后就会触发窗口的 Clicked 事件。若用户单击的是一个可用且可见的控件，就会触发这个控件的 Clicked 事件，而不会触发窗口的 Clicked 事件。

Clicked 事件有 3 个参数，Flags、Xpos、Ypos。Flags 表示用户单击鼠标左键的同时，是否还单击了鼠标上别的键或按了键盘上的键，Xpos、Ypos 分别代表鼠标单击的 X 坐标和 Y 坐标。它只有唯一的返回值 0，表示继续操作。

窗口的 Clicked 和 DoubleClicked 事件被触发之前，先触发 MouseUp 和 MouseDown 事件。

在触发 DoubleClicked 事件之前，先触发 Clicked 事件。

5.4.4 Close 事件

关闭窗口时触发该事件，通常将一些善后的事情放在这个事件中完成，用来消除窗口创建的所有对象。

这个事件没有参数，只有唯一的返回值 0，表示继续操作。

5.4.5　CloseQuery 事件

关闭窗口时，在 Close 事件之前被触发。一般在这个事件中进行安全性检查。

该事件有两个返回值，0 代表可以关闭窗口，继续执行；1 代表不能关闭窗口。默认情况下，这个事件返回 0。但是利用这个返回值的特性，我们可以做许多工作。比如在这个事件中做一些合法性的检查，如果不能通过，就可以通过返回 1 来拒绝关闭窗口。

5.4.6　DoubleClicked 事件

当双击窗口上任何没有被控件覆盖的地方就触发该事件。

它的返回值和参数与 Clicked 事件相同。

5.4.7　Resize 事件

如果窗口是可以改变大小的，当窗口的大小被改变时，或者当用户打开一个新窗口时，都会触发 Resize 事件。一般用这个事件来做一些特殊处理，使窗口在不同大小的情况下保持其外观一致。

5.4.8　SystemKey 事件

当插入点不在编辑框中且用户按下【Alt】键或【Alt+其他键】时就会触发该事件。一般用该事件做一些中断性的处理。

5.4.9　Timer 事件

在 Timer()函数被调用一定的时间之后，Timer 事件会被触发。

Timer 事件没有参数，只有一个返回值 0，表示继续操作。

利用 Timer 事件，可以做一些关于时间方面的处理，如改变窗口上的图形及其位置，动感界面的实现等。Timer 事件的触发时间间隔由窗口的 Timer()函数确定。

5.4.10　RButtonDown 事件

当用户在窗口上任一没有被控件覆盖的地方按下鼠标右键，就会触发 RButtonDown 事件。

RButtonDown 事件具有和 Clicked 事件一样的参数和返回值。唯一不同的就是在 RButtonDown 事件中，右键是默认按下的，因此在参数 flags 中，2 是不计算在内的。

在 RButtonDown 事件中通常用来弹出一个弹出式菜单。

5.5　窗 口 控 件

通过修改窗口的属性，可以美化一个窗口。但是这种方法就好比在精工制作一张画布，再好的画布没有图画也是没有意义的。控件是构成 PowerBuilder 应用程序用户界面、完成数据输入/输出的强有力工具。利用控件，我们能够完成许多界面设计任务而无需编写一行

代码。窗口只有和窗口控件结合起来才能发挥作用。因此了解和熟悉各种控件的功能、作用和用法是设计美观的用户界面的前提和基本要求，只有这样，才能根据应用程序的具体要求，迅速构造出操作方便、使用灵活、安全稳健、界面友好的应用程序。

所谓控件，从本质上来说就是系统预定义好的图形对象。控件是构成应用程序界面的重要元素，同时也是显示和输入数据的场所。每个控件都有一个标识，也就是一个控件不同于另一个控件的名称，把控件名称理解成控件的变量名也未尝不可。同一个窗口中，任何两个控件都不能重名，否则系统将给出错误提示。当我们往窗口上放置控件时，系统自动为控件赋予一个唯一的名称，该名称由两部分组成：第一部分是控件的默认名称前缀，第二部分是个 1 到 4 位的数字，该数字保证控件名称的唯一性。默认的控件名并不直观，它没有反映具体控件的具体作用。一般情况下，我们都要对控件重新命名。

如果对控件名的标准前缀不满意，可以在 PowerBuilder 中进行修改。修改之后，如果再添加控件到窗口上，那么 PowerBuilder 自动给控件取的名字就是按照修改之后的前缀来命名的。

如果想要修改默认的控件标准前缀，首先选择【Design】|【Option】命令，这时就会弹出【Options】对话框。从该对话框中选择【Prefixes 1】或者【Prefixes 2】标签，就可以看到各种控件对象所使用的标准前缀，在这儿也可以对这些标准前缀进行修改。【Options】对话框的【Prefixes 1】标签和【Prefixes 2】标签如图 5.7 所示。

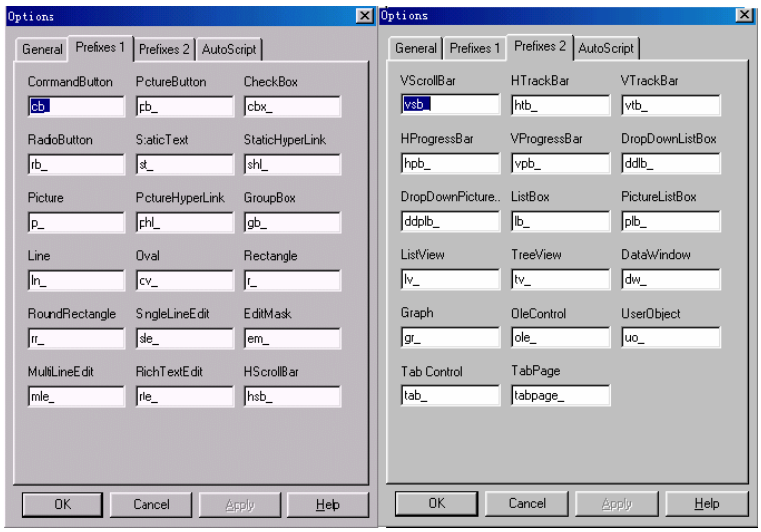

图 5.7 【Options】对话框的【Prefixes】标签

5.5.1 向窗口添加控件

1. 添加控件

要在当前窗口中添加一个控件，可以采取以下两种方法。

1) 通过菜单样式

选择【Insert】|【Control】命令，打开窗口控件列表框，选择需要的控件，在窗口上要添加控件的位置，单击鼠标左键，再调整控件大小、位置、字体等属性即可完成，如

图 5.8 所示。

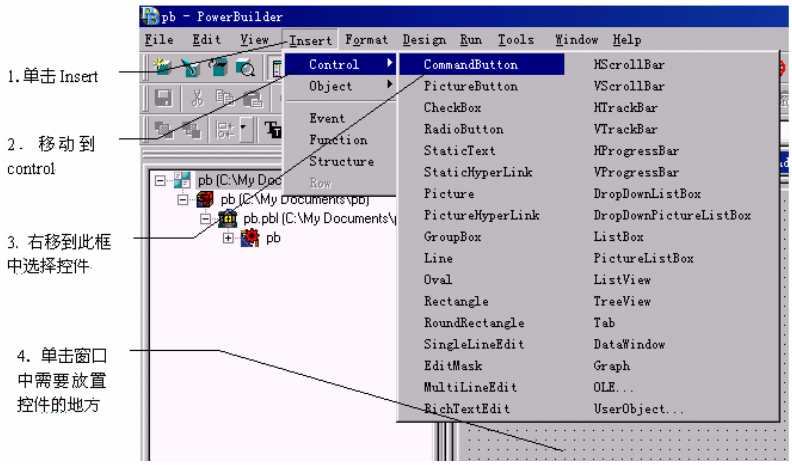

图 5.8　通过菜单添加控件

2) 通过图标按钮方式

单击向下小三角的窗口控件组合图标，弹出窗口控件图标对话框，然后单击需要选择的控件图标，移动鼠标到窗口上准备添加控件的地方，单击鼠标左键，如图 5.9 所示。

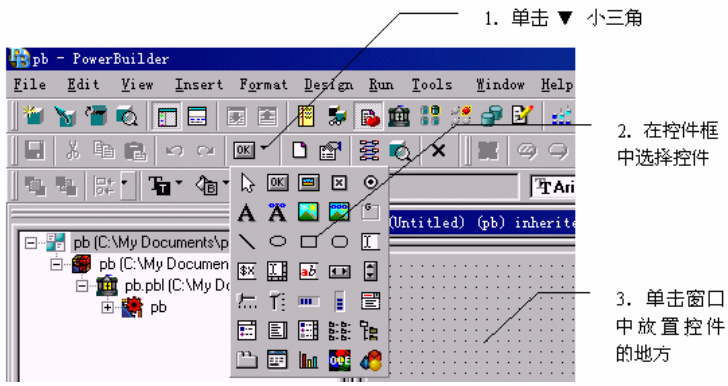

图 5.9　通过图标向窗口中添加控件

当把控件添加到窗口中之后，接下来的事就是如何对控件进行操作。下面简单介绍选择、移动、复制、删除和改变大小这几种操作。

2. 选择控件

1) 选择单个控件

把鼠标移到控件上，单击鼠标左键，则控件的四个角上出现四个实心小方块，表示选择了这个控件，同时在系统状态栏上显示了选择的这个控件名称。此处的四个实心小方块称为把柄。

2) 选择相邻的多个控件

相邻控件是指包含在同一矩形区域中的一组控件，可按以下方法选中它们。

(1) 在这些控件的左上角，按住鼠标左键不放。

(2) 在窗口中拖出一个矩形区域，保证所需控件都在矩形区域内。

(3) 放开左键，此时所有控件都带有把柄，表示这些控件都被选中。

在多个控件被选中后，系统的状态栏显示 Group Selected。

3) 选择不相邻的多个控件

首先选中某个控件，按下【Ctrl】键不放，同时用鼠标单击其他控件，最终所有单击到的控件被选中。若在按下【Ctrl】键的同时，在被选中的控件上单击鼠标左键，则取消选中该控件。

4) 选中所有控件

按下【Ctrl+A】键，或选择菜单【Edit】|【Select All】命令，窗口中的所有控件都将被选中。

3. 删除控件

首先选择要删除的一个或一组控件后，再选择下列方法之一删除选中的控件。

(1) 按下【Del】键。

(2) 选择菜单【Edit】|【Delete】命令。

(3) 单击鼠标右键，选择【Delete】菜单项。

4. 调整控件的位置和大小

1) 移动控件

选择控件后，拖动到新位置。

2) 改变控件大小

选中控件后，移动鼠标经过控件的边界，使它变为双箭头的线条形状↕或↔，然后按住鼠标左键不放并拖动它。

3) 网格

系统提供了网格来帮助调整控件的大小和位置。

(1) 选择【Design】|【Option】命令，打开网格设置对话框。

(2) 在 Alignment Grid 中设置网格的宽度、长度等。

① Snap to Grid：选中该项，当放置和移动控件时，自动滑落到网格点上。

② Show Grid：选中该项，在工作区中显示网格。

③ X：用像素表示的网格中，每个单元的宽度。

④ Y：用像素表示的网格中，每个单元的高度。

5.5.2 设置控件的属性

控件属性的设置决定了控件在运行期间的行为与外观。控件类型不同，属性也不尽相同，但不同类型的控件也有相同的属性。

1. 给控件命名

在窗口中添加一个控件后，系统自动为这个控件生成一个唯一的名称，该名称由默认的前缀和数字组成。

为了使脚本更具可读性，在命名控件时使用系统给定的前缀，而后半部分由用户自定

义，应尽量使用能代表控件功能的单词。

给一个控件命名的基本步骤如下。

(1) 进入该控件的【Properties】视窗。

(2) 选择【General】标签页，在【Name】文本框中显示了系统默认的控件名称。

(3) 留用前缀，删除其他，然后输入有意义的控件名称后缀。后缀可以使用任何有效的标识符，最多 40 个字符。

2. 改变控件文本

1) 通过属性设置

控件【Text】文本框中的内容代表显示在控件上的文本。用系统提供的【Stylebar】工具栏来改变控件的文本，包括文本本身、字体、大小、对齐方式等，命令按钮文本的对齐方式只能是居中。

2) 通过脚本控制

若控件在程序运行的不同时刻，要求显示不同的文本，要实现这一功能就要用脚本进行控制。

如：命令按钮 cb_1，在程序运行时要显示【退出】。可编写脚本：cb_1.text=【退出】。

字体大小使用 textsize 来描述。如 cb_1.textsize=-20，即将字号设置为 20。

3. 定义加速键

为了快速访问一个控件，可以给它定义加速键，这样用户只要按下【Alt+加速键】就能把焦点切换到相应的控件上。

(1) 为【CommandButton】、【CheckBox】、【RadioButton】控件定义加速键。

在控件【Text】属性文本框中在作为加速键的字母前面加上&符号即可，控件以下划线的方式显示加速键。

(2) 为【SingleLineEdit】、【MultiLineEdit】、【ListBox】、【DropDownListBox】控件定义加速键进入控件的【Properties】视窗，在【General】标签页上的【Accelerator】文本框中输入加速键字母。

4. 控件的可访问性

控件有两个属性影响其可访问性，只有这两个属性都设置为真时才能访问控件。

(1) Visible：该属性被选中，则该控件显示在窗口中；该属性未被选中，则程序运行时该控件不显示。系统默认情况下隐藏的控件不在画板中显示，要在画板中显示这些隐藏的控件，通过选择菜单【Design】|【Show Invisibles】命令来完成。

有两种方式改变控件的 Visible 属性。

① 属性设置：选择或取消控件的 Visible 属性。

② 脚本控制：控件名称.visible=True　表示显示控件，

控件名称.visible=False　表示隐藏控件。

(2) Enabled：该属性被选中，则该控件是活动的；该属性未被选中，则在程序运行时该控件只能显示但不能响应键盘和鼠标操作，呈灰色。

有两种方式改变控件的 Enabled 属性。

① 属性设置：选择或取消控件的 Enabled 属性。

② 脚本控制：控件名称. Enabled =True 表示激活控件，由此变得可以使用，
控件名称. Enabled =False 表示控件由此变得不能使用，呈灰色。

5.5.3 调整控件的布局

PowerBuilder 提供了对窗口上控件进行调整的工具。可以根据需要，方便地将窗口布局调整得整齐美观、方便易用，大大地提高了应用程序的设计效率。下面介绍具体的调整工具和使用方法。

1. 齐整性操作

如果通过鼠标拖动的办法来调整控件的位置，使其大小一致、位置整齐，将是非常困难和耗时的工作。为了解决这个问题， PowerBuilder 专门提供了进行齐整性调整的工具。比较常用的方法是利用系统工具栏上的齐整性操作组合图表，共有 11 种齐整性操作图表，具体作用如图 5.10 所示。需要说明的是各种齐整性操作均以第一个选中的控件为基准的。

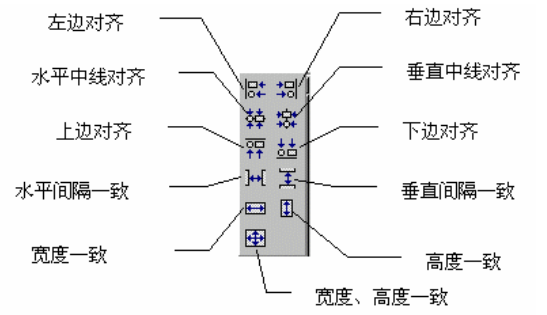

图 5.10　齐整性操作图标

要将一组控件对齐，首先要选择基准位置上的控件。如要使控件左对齐，先选择左对齐基准位置上的一个控件，再选择其他控件，进行左对齐；若要使一组控件之间的上下间距相等，则先选择两个控件，它们相距符合基准间距要求，再选择其他控件，进行相应的对齐设置。对齐控件也可以选择【Format】|【Align】或【Space】或【Size】命令，然后选择对齐方式。其中菜单【Align】弹出控件位置对齐的选项；菜单【Space】弹出调整控件间隔的选项；菜单【Size】弹出调整控件大小的选项。

对齐方式有 6 种：左对齐、右对齐、水平中心对齐、垂直中心对齐、顶部对齐、底部对齐。

2. 窗口控件的 Tab 键顺序

当按下 Tab 键时，窗口中具有操作性的控件会按照一定的顺序改变焦点。合理的顺序对于加快数据输入、方便操作是十分重要的。PowerBuilder 会自动根据操作性控件的位置设定顺序。自动提供的顺序不一定能满足实际的需要，而且布局调整后，自动顺序一般还会改变，这时可以进行手动调整。

首先选择【Format】|【Tab Order】命令，这时，每个控件的 Tab 键顺序号都以红色数字标注在控件的右上角。静态文本类的非操作性控件顺序号为 0，表示得不到活动焦点。

其余控件顺序号从 10 开始，以 10 递增。选中某个控件，即可以对其顺序号进行修改。按照要求的顺序修改序号，修改完成后，再次选择【Format】菜单栏中【Tab Order】菜单项，就完成了 Tab 键顺序的设置。操作过程如图 5.11 所示。

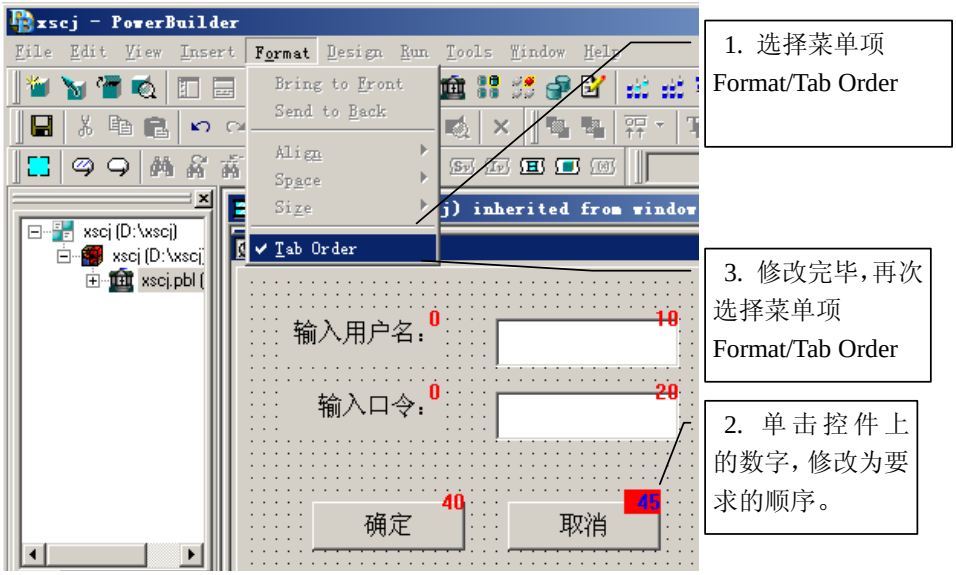

图 5.11　Tab 键顺序的设置过程

需要说明的是：在输入顺序号时，应处于英文状态，若处于中文输入状态，则会造成无法修改顺序号。

5.6　常用的窗口控件

窗口控件除了前面介绍的通用属性外，还有一些与具体的控件类型有关的特殊属性以及事件和函数。下面结合各种控件的编程对窗口控件做进一步介绍。

5.6.1　图形类控件

图形类控件可以用来在窗体上显示简单的图形。PowerBuilder 共提供了四种图形控件，它们分别是直线 Line、椭圆 Oval、矩形 Rectangle 和圆角矩形 RoundRectangle。它们的主要作用只是用来装饰窗口的。下面以矩形为例做一个简要的介绍。

1. 属性

矩形的属性表如图 5.12 所示。

(1)【FillColor】下拉列表框，用来选择填充色。

(2)【FillPattern】下拉列表框，用来指定不同类型的交叉影线。

(3)【LineColor】下拉列表框，用来指定轮廓线的颜色和填充模式。

(4)【LineStyle】下拉列表框，用来指定线或者轮廓线的宽度。

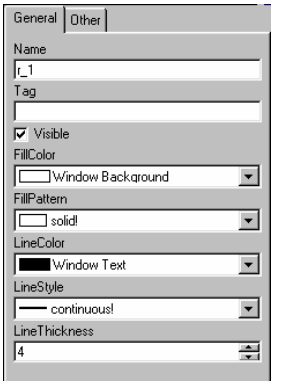

图 5.12　矩形的属性表

2. 事件

图形类控件没有任何事件，也不需要对它们编写脚本。

5.6.2　静态文本控件与图片控件

1. Static Text(静态文本)

静态文本框是用来向用户显示信息的，一般用来显示标题或提示信息等。静态文本之所以称为静态，是因为在程序运行过程中，用户不能手工修改静态文本框中的内容，但可通过脚本改变其显示的内容。该控件在运行时不能获得焦点，即使将它的 Tab Order 值设为非 0 也无法获得焦点。

1) 属性

静态文本的属性如图 5.13 所示。

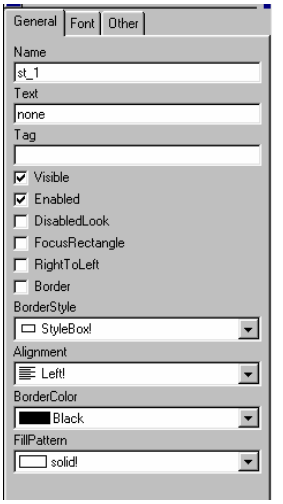

图 5.13　静态文本的属性表

静态文本的属性几乎没有什么特殊的。当要在代码中修改静态文本的内容，就可以通过改变它的 Text 属性来改变它的内容，修改方式是：控件名.text=“新文本内容”。

2) 事件

尽管静态文本拥有很多事件，但是大部分都没有什么实际用途，一般不在静态文本框中编写脚本。

2. Picture (图片控件)

Picture(图片控件)也是一种静态控件，它用来在窗口上布置一幅画来美化界面，以及进行形象说明等。

图片文件名称可以直接在 PictureName 编辑框中输入，也可以按其右侧的“…”按钮在文件窗口中选择。可以直接使用的图片类型有 5 种：“.bmp”、“.jpg”、“.jpeg”、“.rle”和“.wmf”。

图片控件的常用事件是 Clicked，但一般来说图片控件的事件里不需要编写代码。我们有时用图片控件来代替命令按钮和图像按钮，这时，就需要给 Clicked 事件编写代码了。

5.6.3 单选按钮、复选框与分组框

使用图形界面的目的是为了在直观明了的同时方便用户的操作。通过简单的选择就能完成任务，无需用户费力劳神一个个地键入。单选按钮、复选框正是为了方便用户做出选择而设计的。

单选按钮用来表示一组互斥的选项，用户只能从中选择一个。单选按钮带有一个圆形图案，当被选中时，其图案中心出现一个黑点；未被选中，其图案中心为空白。

复选框用来表示“是/否”或者“真/假”两种状态，它常常被成组使用，同一组中可有多个复选框被选中。复选框被选中时，方框内出现一个叉号(×)；或对号(√)没有被选中时，方框内为空白。

单选按钮和复选框的分组是依靠分组框来实现的，因此在常见的界面中，无论是单选按钮还是复选框，它们通常都是和分组框结合在一起的。

1. RadioButton(单选钮)

单选按钮的使用肯定不是单独的，通常是几个单选按钮分成一组使用。我们既可以把多个单选按钮放在一个窗口中(这时窗口中的所有单选按钮成为一组)；也可以用分组框把单选按钮分成不同的组。如果没有使用分组框，那么整个窗口内的单选按钮默认定义为同一组。同一组内的单选按钮彼此之间互斥，也就是说用户只能选择其中的一个。当用户选择了其中的一个单选按钮之后，同组中的其他选中的单选按钮自动取消选中状态。

可以将一组单选钮全部放在窗口中，运行脚本时，则只能有一个单选钮能被选中；若希望窗口中的单选钮能同时选择几个，就要将单选钮分组，每个组中只能有一个单选钮被选中。就是说单选钮选择的个数是根据组数来确定的，若没有分组，则认为是一组，只能选择一个单选钮。

单选钮可以设置为默认选中状态，也可以用脚本设置，被选中的单选钮中间有一圆点。

(1) 设计时设置：选择单选钮进入【Properties】对话框，如图 5.14 所示选择【Checked】选项。

(2) 用脚本设置：控件名.checked=True，表示选择该控件。

控件名.checked=False，表示取消选择该控件。

一般只对单选钮的 clicked 事件编写脚本。

2. CheckBox (复选框)

复选框用于让用户设置独立的选项，多个复选框彼此相互独立，即使把它们放在一个组框中，也不影响复选框的行为，它们依然彼此独立。因此一般情况下，并不使用分组框来约束复选框。

复选框的使用一般也是几个组合在一起，用来表示一些并列但是各自独立的选项，各个选项的选择与否都不影响另外的别的选项。如果需要，还可以默认设置其中的每一个复选框的选中或不选中状态，复选框的属性页如图 5.15 所示。

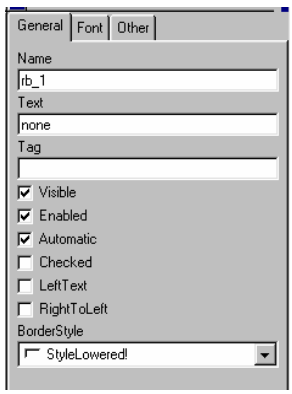

图 5.14　单选按钮的属性页

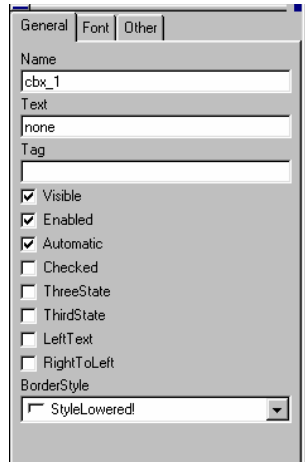

图 5.15　复选框的属性页

复选框可以有 3 种状态，通常使用两种典型状态：选中(【Checked】选项是 True)和未选中(【Checked】选项是 False)。

复选框之间相互独立，可以选择一项或多项，无论是否将复选框分组都一样，若被选中，其方框内有一对号；若未选中，方框内为空。

复选框的事件和单选钮事件基本相似。

3. GroupBox(分组框)

分组框一般有两个用处：表面的用处是美化界面，实际的用处是对单选按钮进行分组。分组框的实际用处只对单选按钮有效，它可以把一个窗口内的单选按钮分成几组。对于复选框，使用分组框是没有意义的，因为复选框本来就不是互斥的，不管用不用分组框，其效果都是一样的。从这个意义上来说，分组框对于复选框只有美化界面的意义，而没有分组的意义。

分组框的属性中：【Text】编辑框用来显示分组框上边界中的标题。可以改变字体类型、字号和字体样式，但是不能改变对齐方式和标题位置，默认为左对齐。

事件：分组框只是用来把一部分控件放在一起，因此它的事件极少，一般也不用为它编写脚本。

5.6.4　编辑类控件

编辑类控件通常用来输入、编辑、显示文本。编辑类控件分为：SingleLineEdit(单行编辑框)、MultiLineEdit(多行编辑框)、EditMask(掩码编辑框)等。

1. SingleLineEdit(单行编辑框)

单行编辑框中只能输入或输出一行文本信息，有时候也用来输入一些保密信息。单行编辑框专门提供了一个属性 Password，用来屏蔽用户输入时的回显，比如让用户输入密码等。

2. MultiLineEdit(多行编辑框)

多行编辑框一般用来输入大量的文本，如员工的简历、大段的文章。输入或输出多行数据。

单行编辑框和多行编辑框共同的属性如下。

(1) Password 选项：一般选择该项是为了在输入密码时，不让输入的信息在屏幕上显示出来，屏幕上显示的只是星号(*)。

(2) AutoHScroll：选择该项可使用户输入更长的字符串，当输入的字符超过右边界时，文本会自动向左滚动。

(3) DisplayOnly：选择此项只能显示数据，不能输入数据。

(4) TextCase：限制控件只能接受大写字母(upper!)、小写字母(lower!)或大小写都可以(anycase!)。如选择“lower!”，则只能接受小写字母，若输入了大写字母，则自动转换为小写字母。

(5) Limit：控件中可以输入的最多字符数。0 表示不限制输入的字符数。

(6) HScrollBar 和 VScrollBar：在控件内部显示水平滚动条和垂直滚动条。

(7) AutoVScroll：当输入的行数超过屏幕所能显示的行数时，控件内的行自动向上滚动。

(8) IgnoreDefaultButton：当用户按下回车键时，不触发 Default 按钮事件，即使该按钮被定义为默认按钮。

(9) Modified 事件：当用户修改了编辑框(单行、多行编辑框)中的文本，按下了键盘上的【Tab】键、回车键，或者用鼠标把输入焦点转移到别的控件对象时，就会触发编辑框的 Modified 事件。

(10) GetFocus 事件：编辑框获得输入焦点之时就会触发 GetFocus 事件。

(11) LoseFocus 事件：编辑框失去输入焦点时就会触发 LoseFocus 事件。

3. EditMask(掩码编辑框)

用户要输入固定格式的数据时，就可以使用掩码编辑框。用户只能输入指定格式的数据，使输入规范化，避免输入错误，同时可以提示输入数据的格式。

掩码编辑框控件的主要属性在【Mask】标签页中设置，如图 5.16 所示。

掩码数据有 6 种类型：日期型、时间型、日期时间型、数值型、数字型、字符串型。

在使用掩码编辑框时，有时要用到以下选项。

(1) Mask：决定数据输入的格式，可以输入 Mask 选项或空白。

(2) Spin：选择该项在编辑框的右边显示一个上下箭头(微调按钮)，单击上下箭头数据按指定的间隔增大或减小。

(3) Increment：单击微调按钮数据之间的间隔。

(4) Min、Max：微调按钮输入数据的变化界限。

(5) UseCodeTable 和 DisplayData：这两个属性结合使用，它们只在【Spin】选项被选中时才能起作用。它们表示输入的数据是有规律的，按固定表中的数据选择输入。

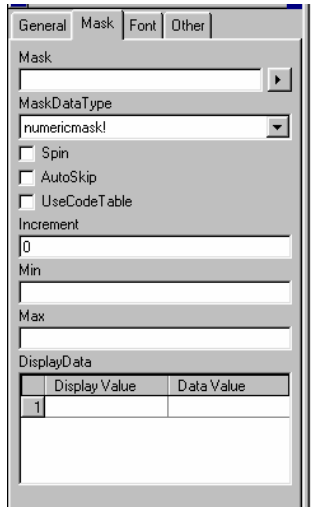

图 5.16　掩码编辑框控件的【Mask】标签页

掩码编辑框的事件有十几个，最重要的事件有 Modified 事件、GetFocus 事件和 LoseFocus 事件，事件的触发类似于单行编辑框。

5.6.5　按钮类控件

PowerBuilder 中的按钮与现实生活中的按钮十分相似，当用户单击(或者说“按下”)按钮时，将会触发一个操作。PowerBuilder 标准的按钮控件一共有两种，分别为 CommandButton(命令按钮)和 PictureButton(图形按钮)。这两种按钮的区别并不是很大：命令按钮是 Windows 的一种标准按钮，具有文本标题；而图形按钮除了具有标题之外，还可以指定显示在按钮上的图形文件。

1. CommandButton(命令按钮)

命令按钮总以三维形象显示，它没有其他大多数控件具备的 Border(边框)属性，我们也不能修改按钮的标题字符颜色和背景颜色。当用户单击按钮时，命令按钮会呈现按下状态。

命令按钮的属性如图 5.17 所示。

可以把窗口中的一个命令按钮设定为默认按钮，当焦点不在任何一个命令按钮上，只要按下回车键或【Esc】键就自动执行按钮上的 Clicked 事件。方法如下。

打开按钮属性对话框，选择【General】标签。

(1) 选择【Default】选项，就能实现按下回车键相当于单击命令按钮，该按钮被称为【Default】按钮。

(2) 选择【Cancel】选项，就能实现按下【Esc】键相当于单击命令按钮，该按钮被称为【Cancel】按钮。

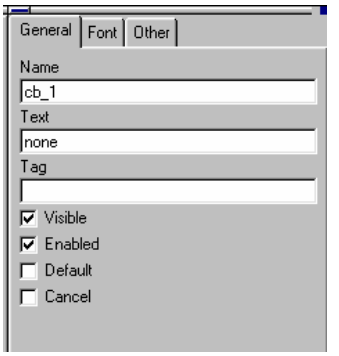

图 5.17　命令按钮属性页

一组命令按钮中只能有一个被指定为【Default】按钮和【Cancel】按钮。

若窗口中有两个按钮，分别是“确定”、“取消”，一般将“确定”按钮设定为【Default】按钮，将“取消”按钮设定为【Cancel】按钮。

命令按钮有十几个事件，不过一般情况下只使用它的 Clicked 事件。这个事件是在用户单击这个命令按钮并且释放了鼠标左键时触发。如果在单击这个命令按钮之前，当前窗口的输入焦点不是在这个命令按钮上的，那么会同时触发这个命令按钮的 GetFocus 事件，当前命令按钮就会获得输入焦点。

2. PictureButton (图形命令按钮)

图形命令按钮的使用与命令按钮相似，但它可在按钮上指定一幅或两幅图片。把图片放在图形命令按钮上的方法如下。

(1) 进入该图形命令按钮的【Properties】对话框，选择【General】标签。

(2) 在【PictureName】文本框中确定为 Enable 时的图片文件。

在【DisableName】文本框中确定为 Disable 时的图片文件。

5.6.6　列表类控件

列表类控件是提供给用户一组可选的或可视的选项，用户可以选择其中的一项或多项。列表类控件一共有 4 个，它们分别是列表框、图形列表框、下拉列表框和下拉图形列表框。

1. 列表框

列表框是在一个可滚动的矩形框内显示多行文本，用户可以选择其中的一行或多行。

1) 常用的属性设置

(1)【General】标签页中，属性设置如下。

【Sorted】复选框：用来决定列表框中各项文本是否按顺序重新排列。

【HScrollBar】和【VscrollBar】选项：决定是否在需要时显示水平滚动条和垂直滚动条。

【ExtendedSelect】复选框：决定是否可以一次选择多行。多行选择可通过鼠标拖动、选择一项后再按住【Shift】键选择另一项，最终选择一个连续区域、或选择一项后再按下【Ctrl】键选择不连续的选项，3 种方法均可。

【MultiSelect】复选框：决定是否可以通过鼠标单击的方法选择多项，而不需要用【Ctrl】键或【Shift】键同时操作。

(2)【Items】标签页中的列表项设置。

Item 用于指定列表框中显示的列表项。可以在运行前事先指定，也可通过脚本插入列表项，插入方法如下。

```
列表框名称.insertitem(文本行字符串，index)
```

该语句表示在列表框的第 index 行之前插入一项内容为“文本行字符串”，系统给所有列表项一个唯一的索引 index，用它来区分各个列表项。

2) 常用事件介绍

(1) SelectionChanged 事件。

用户选择了列表框中的一个列表项之后，就会触发该事件，若通过鼠标完成项目的选择，则还会触发列表框的 Clicked 事件。

SelectionChanged 事件只有一个参数 index，它是一个整数，用来表示新选择的列表项的索引号。

SelectionChanged 事件只有一个返回值 0，表示继续操作。

(2) DoubleClicked 事件。

当用户双击列表框中的列表项时，会触发DoubleClicked 事件，同时还会触发 SelectionChanged 事件。

该事件的参数及返回值同 SelectionChanged 事件。

2. 图形列表框

图形列表框与列表框基本相同，唯一的区别是列表框中的列表项只有文本显示，而图形列表框的列表项除了显示文本外，还在文本之前加了一个小图标。

图形列表框的属性视窗比列表框多了一个【Pictures】标签页，在【Items】标签页中多了一个 PictureIndex 项。

3. 下拉列表框

下拉列表框实际是单行编辑框和列表框的组合。与列表框相比，它有两大好处：一是它的列表项可以作为一个单行编辑框，二是它节省了屏幕空间。

下拉列表框有两种类型，不可编辑的和可编辑的。对于不可编辑的，用户只能选择列表框中的列表项；对于可编辑的用户除了可以选择列表框中的列表项外，还可在列表框的编辑框中输入列表项。

1) 属性设置

【AllowEdit】复选框：决定下拉列表框是否是可编辑的。选择该项表示允许编辑，未选中该项表示只能选择信息不能编辑信息。

【ShowList】复选框：决定是否一直显示列表框和下拉箭头。为了节约屏幕空间，可以不选择这个复选框；若不选择该项表示只有用户单击下拉列表框时，才会显示下拉箭头，

才会显示列表项。

2) 事件简介

下拉列表框的事件是列表框和单行编辑框事件的组合。其中最重要的事件是 SelectionChanged 事件、Modified 事件和 DoubleClicked 事件。

SelectionChanged 事件：只有用户从下拉式列表框中选择列表项时才会触发，在编辑框中进行输入不会触发这个事件。

Modified 事件：在下拉列表框处于可编辑状态下才能触发；否则就不能触发。触发顺序为：先 SelectionChanged 事件，后 Modified 事件。

DoubleClicked 事件：在下拉列表框的 ShowList 属性为 True 时才会被触发，否则不会触发。

4. 下拉式图形列表框

下拉式图形列表框可以显示图标，它的属性视窗比下拉式列表框多了一个【Pictures】标签页。

5.6.7　选项卡控件

选项卡控件放置在窗口中，选项卡控件中的选项页也可以放置命令按钮、数据窗口等控件。选项卡控件都包含有一个或多个选项页，每个选项页由两部分组成，一部分是放置选项页标题的标题部分，另一部分是放置选项页的区域。每一时刻用户只能看到一个选项页，用户看到的选项页就是当前使用的选项页，这个选项卡页的标题会凸出来，其他选项页的标题会凹下去。

1. 创建一个选项卡

在下拉控件按钮中，选择选项卡控件，在窗口空白位置上单击，就可创建一个选项卡。新创建的选项卡中有一个选项页，选项页的标题是 none，如图 5.18 所示。一般情况下，选项控件都要使用多个选项页，为此需要在选项卡控件中插入选项页。

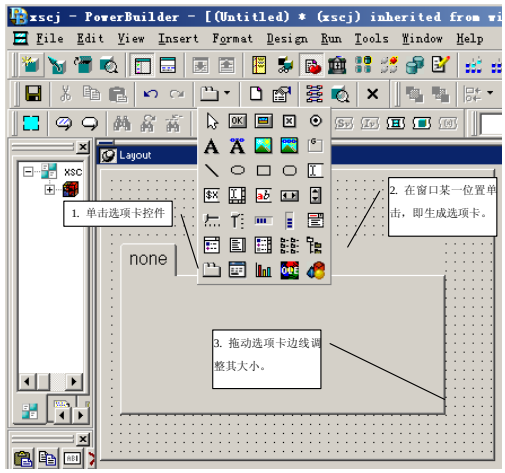

图 5.18　创建选项卡

2. 在选项卡中插入一个选项页

在选项卡的标题部分右击，在弹出的菜单中选择【Insert TabPage】命令即可，如图 5.19 所示。

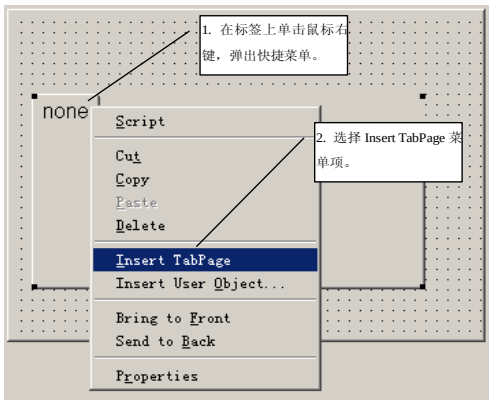

图 5.19　插入选项页

3. 选项卡及选项页属性

选择选项页的标题右击，选择【Properties】菜单项，显示选项卡的属性；选择选项页区域右击，选择【Properties】菜单项，显示的将是选项页的属性。

选项卡及选项页常用的属性如下。

1)【General】选项卡

【Name】文本框：输入并确定选项卡名称，该名称用于程序设计。

【PowerTips】复选框：要与 TabPage 选项页中的 PowerTipText 选项结合使用。当后者输入文本内容，且前者被选中，则当鼠标移至选项页标题时，自动显示在 PowerTipText 中定义的提示内容。

【FixedWidth】复选框：决定选项卡是固定宽度，还是随其上面的文本长度自动调整宽度。选择该项，宽度固定，否则宽度可变。

ShowText 和 ShowPicture：指定选项页的标题上是否可显示文本和图形。选中显示，否则不显示。

【Perpendicular Text】复选框：决定标题文本是垂直显示还是水平显示。选择该项则是垂直显示。

【MultiLine】复选框：决定选项卡的标题文本是否可以多行显示。

【BoldSelectedText】复选框：决定是否用粗体显示当前选中的选项卡页标题。

【SelectedTab】列表框：指定初始显示的选项页。如选择 2，则程序运行时，默认指向第二个选项页。

【TabPosition】列表框：用来指定选项卡显示的位置。

在【PageOrder】选项卡中显示目前选项页排列的顺序，若要改变当前顺序，可以拖动选项页完成。

2)【TabPage】选项卡

TabText 文本框：输入选项页标题文本。PowerTipText 文本框：其文本与选项卡 General

属性的 PowerTips 复选框结合使用。当前者输入文本，且后者被选择中，当鼠标移至选项页标题，片刻自动显示在 PowerTipText 中定义的提示内容。PictureName：设定选项页的图形，该项与选项卡的 General 属性的 ShowPicture 复选框结合使用。当前者指定了图形，且后者被选中，则选项页上会显示图形。

4. 选项卡事件

选项卡事件较多，但常用较少，下面介绍常用的选项卡事件。

1) SelectionChanging 事件

当用户选择一个新的选项页时，首先触发 SelectionChanging 事件，再触发 SelectionChanged 事件。SelectionChanging 事件是在改变选项页时触发，而 SelectionChanged 事件是在选项页已经改变之后触发。

SelectionChanging 事件有两个参数，即 oldindex 和 newindex，它们都是整型参数。oldindex 表示原来选中的选项页的索引号，newindex 表示新选择的选项页的索引号。

SelectionChanging 事件有两个返回值：0 和 1。若返回 0 表示继续操作；若返回 1 表示拒绝操作，此时因操作被拒绝，新选择的选项页不能成为当前选项页。

一般利用 SelectionChanging 事件的返回值，实现应用程序中合法性检查的内容。如在一个选项卡失去焦点之前，检查它的控件的值是否符合要求，不符合要求返回 1 阻止焦点转换。

2) SelectionChanged 事件

一般用来初始化用户新选择的选项页，其参数及返回值与 SelectionChanging 事件相同。

5.6.8　数据窗口控件

数据窗口控件是一个标准的 PowerBuilder 控件。数据窗口控件是用来放置数据窗口对象的，它能用多种表现风格去显示数据。该控件可以显示从数据库中提取的数据，允许用户对数据进行编辑、修改、增删等操作，还能够进行过滤、查询、打印等，是一个功能非常强大的控件。

数据窗口控件作为一个标准控件，也有它自己的属性表。如图 5.20 所示。

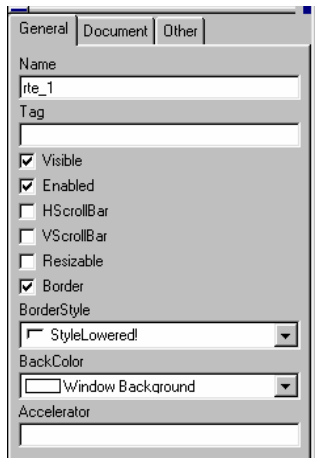

图 5.20　数据窗口控件属性表

【DataObject】编辑框：用于输入和当前数据窗口控件相连接的数据窗口对象的名字。用户也可以不从键盘输入，而是通过单击编辑框右边的【…】按钮进行浏览，从而指定相应的数据窗口对象。

【LivingScrolling】复选框：决定当垂直滚动条上的滑块被上下拖动时，数据窗口对象中的数据行是否也随着动态滚动。一般情况下它比较有用。但是表中的数据行比较多时，选中这个复选框会使上下拖动滚动条的速度变慢。

【HSplitScrolling】复选框：用来决定是否将数据窗口对象分成并列的两个数据窗口对象。这两个并列的两个数据窗口对象之间的界线是可以移动的，而且可以独立地对这两个数据窗口对象分别进行滚动显示。

【RightToLeft】复选框：决定是否把数据窗口控件内的数据从右向左地显示。

【Resize】复选框：决定用户是否可以调整窗口的大小。

【Border】复选框：决定是否有边框。而【BorderStyle】下拉式列表框则用来决定当前数据窗口控件的边框样式。

5.6.9 用户对象控件

用户对象是用户自己利用 User Object 画板制作的特殊控件。它通常是很多控件和代码的组合体。

1. 属性

从【Select Control】下拉式图标中选择用户对象控件，弹出对话框，如图 5.21 所示。

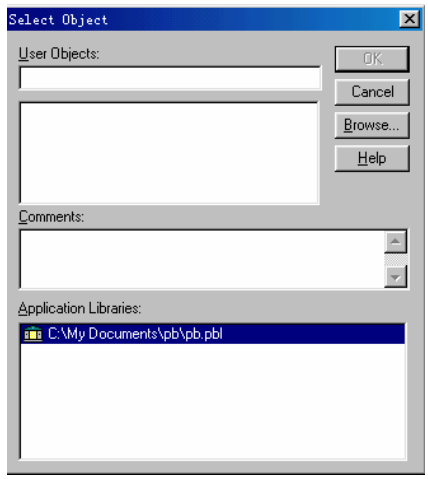

图 5.21 【Select Object】对话框

从这个对话框中选择要作为控件添加在窗口上的用户对象，然后单击【OK】按钮就可以了。把控件添加到窗口上之后可以看到该控件的属性表，这个属性表比较简单，没有任何特殊的选项。

2. 事件

用户对象根据类型的不同，所拥有的事件也不同。关于用户对象的事件信息，在本书

的第 8 章会详细介绍。

5.6.10　常用的窗口控件编程实例

掌握前面对创建窗口的描述后就可以进一步完善学生成绩管理系统应用程序了。在对学生成绩管理系统进行完善的过程中，将用到前面介绍的关于窗口和 Windows 画板的一些基本知识。以登陆界面为例，如图 5.22 所示，介绍一下创建 Windows 窗口的过程。

图 5.22　登陆界面

1. 建立用户界面的对象

首先明确这个窗口的显示形式，比如有哪些控件，对控件进行操作发生哪些事件，控件间的关系等。在登陆界面中，有 3 个静态文本编辑器；两个单行文本编辑器 sle_1、sle_2，可以分别接收用户名和密码；两个命令按钮 cb_1、cb_2，可以分别实现密码确定和放弃登陆的功能。把这些控件添加到窗口上，调整到合适的位置。

2. 对象属性的设置

对象建立好后，就要为其设置属性值，设置属性是为了使对象符合应用程序的需要。在接收密码的单行文本编辑器 sle_2 的 Password 属性中输入“*”，屏蔽输入的密码。

3. 对象事件过程及脚本

建立了用户界面并为每个对象设置了属性后，就要考虑用什么事件来激发对象执行所需的操作。这涉及选择对象的事件和编写事件过程的脚本。

本例中“确定”命令按钮的 Clicked 事件用来接收用户输入的用户名和密码，并和用户表中的记录进行比较，结果正确则进入系统主界面，错误则重新输入，3 次错误退出应用程序。

(1) 首先定义窗口的实例变量：int li_n。

(2) 在窗口的 Open 事件中给变量赋初值：li_n=3。

(3) 在“确定”命令按钮的单击事件中编写如下脚本：

```
string ls_username, ls_password
ls_username=trim(sle_1.text)      //输入的用户名和密码
```

```
ls_password=trim(sle_2.text)
if ls_username="" or ls_password="" then
messagebox("提示","用户名和密码不能为空")
else
SELECT "users"."name", "users"."password", "users"."admin"
    INTO :gs_username, :gs_password, :gs_admin
    FROM "users"
    WHERE ( "users"."name" = :ls_username ) AND
      ( "users"."password" = :ls_password );
  if sqlca.sqlcode=0 then
  open(w_main)            //密码正确，打开主窗口
  close(w_login)
  else
   li_n=li_n-1
   if li_n<>0 then
     messagebox("提示","用户名或密码错误")
   else
      messagebox("提示","错误超过 3 次，自动退出")
      halt                //错误超过 3 次，退出程序
      end if
  end if
end if
```

4. 保存窗口

对设计好的窗口进行保存，单击工具栏中的“保存”按钮保存 Windows 窗口。

5.7 窗口的继承

面向对象编程的特点之一是具有继承性，PowerBuilder 的窗口也具有继承性。有的情况下，我们已经有了一个类似的标准窗口，其他的许多窗口都与这个标准窗口类似。这时候就可以使用窗口的继承，只需要对继承而来的窗口作适当的修改即可，从而可以节省设计和编程的工作量。

在学生成绩管理系统中，有多个录入窗口，包括班级录入、成绩录入、基本信息录入等窗口。这些窗口都是为了实现信息的录入，窗口的样式、控件、事件和脚本等都相同，唯一的区别就是数据窗口控件中的数据不同。这样我们就可以先建立一个名为 w_shuru 的输入窗口，界面如图 5.23 所示，实现数据的添加、删除、保存同时还包括记录的浏览等功能。把此输入窗口作为祖先窗口，通过继承得到班级输入窗口 w_banji_shuru 和基本输入窗口 w_jiben_shuru。

图 5.23　录入窗口

5.7.1　通过继承创建窗口

实现窗口的继承的方法是：选择【File】|【Inherit】命令，或者单击工具栏【继承】按钮，就会弹出选择继承对象的对话框，如图 5.24 所示。首先选择对象类型为窗口，然后在上面列出的窗口对象列表单中选择要继承的祖先窗口，双击鼠标左键或单击后，再单击【OK】按钮确定。这时，与选中窗口具有相同外观的继承窗口就生成了，并显示在窗口画板中，我们根据需要对后代窗口做适当的修改，然后以新文件名存盘。在本例中，我们对数据窗口控件中的数据窗口进行修改，选择班级输入数据窗口，以 w_banji_shuru 为名保存此后代窗口，就得到录入班级信息窗口，如图 5.25 所示。同样的方式我们可以得到基本信息录入窗口。

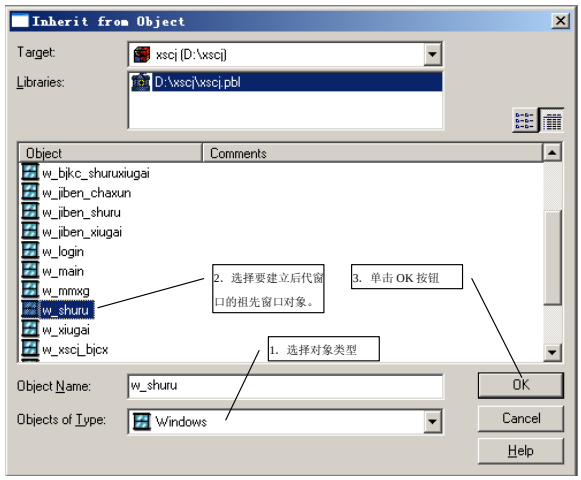

图 5.24　【Inherit from Object】对话框

使用继承法建立窗口的好处主要有下面两点。

(1) 当修改了祖先窗口后，其修改将影响到后代的所有窗口，而不用手工修改每一个后代，这样可以节省设计和编写脚本的时间，应用程序也易于维护。

(2) 由于后代继承了祖先的所有脚本，因此不必重新编写脚本，从而保证了应用程序

中代码的一致性。

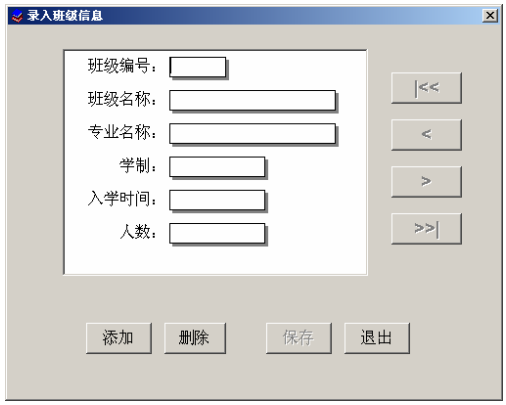

图 5.25　录入班级信息窗口

5.7.2　继承窗口的修改

使用窗口的继承建立窗口对象时，祖先对象中的所有内容，包括窗口、控件、函数、事件以及结构等全部被它的后代继承。在后代窗口中，可以引用祖先的函数、事件以及结构，改变窗口的属性以及窗口和控件的大小和位置，修改现有的控件以及添加新的控件，编写新的脚本，声明新的变量、函数、事件以及结构。

使用窗口的继承，需要注意以下两点。

(1) 后代窗口中所有继承来的控件，都不允许删除。遇到不需要的祖先控件，可以采用将该控件的可视性属性(Visible)不选中，让其在后代窗口中不可见的方法消失。

(2) 祖先与后代窗口中的控件名称必须唯一，不能使用相同的控件名称。

5.7.3　继承窗口的事件脚本

使用继承方法得到的窗口，后代继承了祖先的所有脚本，但继承窗口并不是一成不变的，可以增加控件，新增控件可以有自己的脚本。对于继承来的脚本，也可以对其进行扩展，以满足不同的需求。

5.8　小　　结

创建窗口是创建应用程序中比较基本的工作。当然，对窗口对象的创建大部分都是可视化的，除了代码脚本的编写之外。在窗口上放置的各种控件通过排列和预览，可以得到更好的实际效果。灵活地运用窗口的一些编程技巧，可以使用短小精悍的程序来实现复杂多样化的功能。

没有一个应用程序的用户交互界面是只有窗口的，所有的应用程序界面总是需要包含控件对象。窗口之所以成为用户交互的基本界面元素，最主要的原因是在它上面可以添加各种各样的控件。PowerBuilder 提供了许许多多的控件，它们用来完成不同的功能。在应

用程序中根据不同的需要，可以选择使用不同的控件。熟练使用各种控件可以使应用程序的界面操作更加简洁和方便。

5.9　实　　训

实训目的

(1) 本章将通过上机练习使学生掌握窗口的创建和窗口属性的调整。
(2) 加深对窗口的类型、各种类型窗口的特点以及应用范围的认识。
(3) 掌握常用控件的基本使用方法。

实训内容

(1) 制作一个简单的计算器应用程序。计算器的功能较为简单，效果如图 5.26 所示。

(2) 使用控件：设计一个个人资料输入窗口。使用单选按钮选择“性别”，组合列表框选择“民族”，复选框选择“个人爱好”。当单击【确定】按钮，将个人资料信息输出在“个人资料”的框架内，程序运行界面如图 5.27 所示。

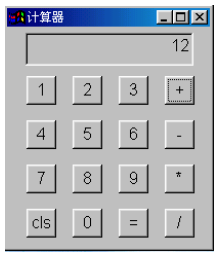

图 5.26　计算器

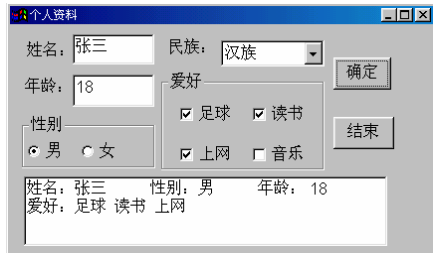

图 5.27　个人资料

(3) 选项卡的应用：在第一个选项页中输入成绩，在第二个选项页中显示成绩及其等级。运行界面如图 5.28 所示。

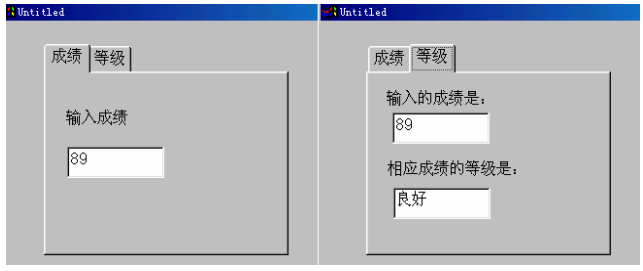

图 5.28　选项卡的应用

实训过程

1. 制作一个简单的计算器

分析：这个计算器实现的功能比较简单，但可以帮助你了解如何添加控件，对控件进

行合理的布局，利用脚本去完成相应的功能。

实训步骤

(1) 建立应用程序，建立 Window 对象，进入窗口画板，在此画板中，可以对窗口进行设置。

(2) 现在开始创建计算器的模型。单击【SingleLineEdit Control】按钮，创建计算器的显示屏，使其 General 属性中的 Enabled 处于未选中状态，这样计算器的显示屏就不能被修改了。然后创建计算器的按键。单击【CommandButton Control】按钮，连续创建从“0”到“cls”，再应用画板中的对齐功能进行对齐。

(3) 在应用画板中，先定义全局变量，在【Declare Global Varibles】窗口中，选择【Global Varibles】按钮，在此窗口中输入如下脚本。

```
double result          //定义计算结果变量，类型为双精度
string buffer= ""      //定义一个输入缓冲区变量
int judge=0            //定义运算类型的判断，0 为“=”，1 为“+”，2 为“-”，3
                       //为“*”，4 为“/”
```

(4) 现在为计算器按键添加脚本。在数字键(0～9)中添加如下脚本。

```
if judge=0 then
   result=0            //result=0 时，此时为输入状态
  endif
  buffer=buffer+"0"    //“0”是数字键为 0 时，如为 2 时应是 buffer+"2"
  sle_1.text=buffer    //显示输入的数字
```

(5) 在运算符按键(+、-、*、/、=)中添加如下脚本。

```
double buf
buf=double(buffer)
CHOOSE CASE judge
CASE 0
     If result=0 then
        Result=buf
      End if
 Case 1
     Result=result+buf
Case 2
     Result=result –buf
Case 3
    If buffer<>"" then
         Result=result*buf
     End if
Case 4
     If  buf<>0 then
         Result=result/buf
     End if
END CHOOSE
If result<100000000 then
   Sle_1.text=string(result)
Else
   Sle_1.text="error"
```

```
End if
Buffer=""
Judge=1        //judge=1 是当按下“*”键时返回的值，返回值在全局变量定义时已给出，
               //其余部分相同
```

(6) 在 cls 按键中添加如下脚本。

```
sle_1.text="0"
buffer=""
judge=0
result=0
```

2. 使用常用控件

分析：用户可以使用颜色和工具条之类的东西来美化窗口的外观，但是控件才是真正具有某些窗口的功能。在运行应用程序时，用户主要通过放置在窗口中的控件与应用程序进行交互。本实例主要演示了如何使用按钮、静态文本、复选框、单选按钮等控件。

实训步骤

(1) 创建应用程序，创建 Windows 窗口。

(2) 添加控件，对控件进行合理的布局。

(3) 添加脚本。因为添加、选择相关内容后单击【确定】按钮才显示相关内容，所以在 CommandButton 控件中添加脚本即可。

3. 选项卡的应用

分析：选项卡的使用使应用程序的使用变得较为简单。本例所实现的简单功能，主要通过本例来介绍选项卡的应用。

实训步骤

(1) 创建应用程序，创建 Windows 窗口。

(2) 添加控件，对控件进行合理的布局。

(3) 添加脚本。在选项卡的第一个选项页中输入成绩后，选择第二个选项页自动显示成绩和成绩的等级，因此在选项卡的 SelectionChanged 事件中添加相应的脚本。

```
tab_1.tabpage_2.sle_2.text=tab_1.tabpage_1.sle_1.text
choose case integer(tab_1.tabpage_1.sle_1.text)
    case 90 to 100
        tab_1.tabpage_2.sle_3.text="优秀"
    case 80 to 89
        tab_1.tabpage_2.sle_3.text="良好"
    case 70 to 79
        tab_1.tabpage_2.sle_3.text="中等"
    case 60 to 69
        tab_1.tabpage_2.sle_3.text="及格"
    case is<60
        tab_1.tabpage_2.sle_3.text="不及格"
end choose
```

通过本章的上机实验，学员应该能够掌握怎样使用窗口控件，对控件进行合理的布局，

添加事件脚本。

5.10 习　　题

1. 窗口画板中有哪些区域，各自有什么用途？怎样打开和关闭这些区域？

2. 窗口有哪几种类型？各自有什么特点？一般应用于哪些场合？

3. 怎样在窗口事件中编写脚本？

4. 什么是函数的静态调用和动态调用？这两种调用方法各有什么优缺点？怎样实现函数的动态调用？

5. 怎样向窗口中添加控件？

6. 如何进行窗口中控件的布局调整？

7. 窗口控件有哪些通用属性？Enabled 属性和 Visible 属性有什么特点？不选中时外观上有什么不同？

8. 什么是窗口控件的快捷键？怎样定义窗口控件的快捷键？

9. 怎样选择不同的选项页？在选项卡控件的什么位置单击时，显示的是选项页的属性？在什么位置单击时，显示的是选项卡的属性？当需要生成新的选项页时，应当在什么位置单击鼠标右键？

10. 制作修改窗口 w_xiugai，要求界面如图 5.29 所示，实现相应的功能。把此修改窗口作为祖先窗口，通过继承得到班级修改窗口 w_banji_xiugai 和基本修改窗口 w_jiben_xiugai。

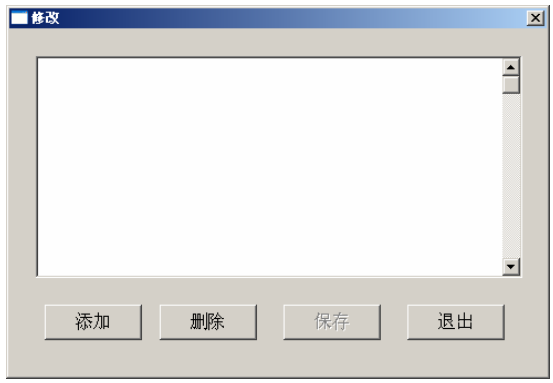

图 5.29　修改窗口