定制 Visual C++ 应用程序用户界面技术

杨少波　（中科院计算技术研究所　100080）

摘要：本文通过实例着重介绍在 Visual C++ 5.0 版应用程序中定制位图菜单、对话框及子控制属性的技术实现，以编制出具有个性化的用户界面的 Windows 应用程序。

关键词：Visual　C++　用户界面　定制

一、概述

微软 Visual C++ 5.0 版 Develop Studio 是一个功能强大的 Windows 应用程序开发平台，用户利用其中的 MFC 类库及 AppWizard、ClassWizard、资源编辑器等工具，可以快速地生成标准 Windows GUI 用户界面的应用程序；但为体现个性化及应用环境要求，用户可能要设计自己特殊属性的用户界面（如定制对话框、子控制及菜单等）。本文通过实例论述这方面的技术实现。

二、用户化对话框及子控制的颜色

在应用程序中为对话框选择特定的颜色，一般只需在派生应用类的 InitInstance() 内修改对 SetdialgBkColor() 的调用即可。如 SetDialogBkColor(RGB(0,255,255), RGB(255,0,0))；但这将使所有对话框的背景色为青色，对话框内的静态文本为红色。

图 1　用户化颜色对话框

如要求只改变个别对话框的背景色及将对话框中的各个子控制颜色定制，则不能采用上述方法，而应在派生对话框类实现 WM-CTLCOLOR 消息响应函数，来实现设定对话框静态文字前景色及对话框的背景色。当然也可以用户化各个子控制的颜色，因为每个控制在显示之前会立刻向其父窗口发送 WM-CTLCOLOR 消息，对话框本身也会发送该消息。图 1 为本文的一个用户化颜色对话框。其背景色为青色，编辑条控制为红色（因文章打印时未采用彩色打印机，故未见到颜色）。

对话框类定义及利用 ClassWizard 产生的 WM-CTLCOLOR 消息响应函数实现如下（其中 m-hCyanBrush 和 m-hRedBrush 在对话框类的 OnInitDialog() 内进行初始化，见下文 3 节）。

```
class CAboutDlg : public CDialog
{ public: HBRUSH m-hCyanBrush, m-hRedBrush; //涂刷成员数据
    CBitmapButton ButtonOK, ButtonCancel; //替换法的位图按钮
    CDrawButton m-icon; // 派生法自绘按钮
    //{{AFX-DATA(CAboutDlg)
    enum { IDD = IDD-ABOUTBOX };
    CEdit m-EditBox; //编辑条控制
    //}}AFX-DATA
protected:
    //{{AFX-MSG(CAboutDlg) //保留给 ClassWizard 用
    afx-msg HBRUSH OnCtlColor(CDC * pDC, CWnd * pWnd, UINT nCtlColor);
    virtual BOOL OnInitDialog();
    afx-msg void OnBigicon();
    ... //其他成员定义
    //}}AFX-MSG
    DECLARE-MESSAGE-MAP()
};
HBRUSH CAboutDlg::OnCtlColor(CDC * pDC, CWnd * pWnd, UINT nCtlColor)
{ HBRUSH hbr = CDialog::OnCtlColor(pDC, pWnd,
```

```
nCtlColor);
    if(nCtlColor = = CTLCOLOR-EDIT) //是编辑条控制?
    { pDC - > SetBkColor(RGB(255,0,0)); //置为红色
      return m-hRedBrush; //返回涂刷
    }
    if(nCtlColor = = CTLCOLOR-DLG) //是对话框?
    { pDC - > SetBkColor(RGB(0,255,255)); //置为青色
      return m-hCyanBrush; //返回涂刷
    }
    return hbr; //返回缺省的涂刷给其他控制
}
```

三、对话框中使用自绘位图按钮

为避免对话框内采用标准的按钮而显得呆板，可以内嵌自绘的位图按钮，其实现途径可以归纳为两种：其一为替换法，即用自绘位图按钮替换采用对话框编辑器生成的标准按钮（见图 1 对话框中的“是”和“取消”按钮）；另一方法则是派生法，也就是以标准的按钮 CButton 类为基类派生自己的自绘按钮类（见图 1 对话框中左上角“电话机”按钮）。

1. 替换法内嵌位图按钮

首先利用对话框编辑器创建自己的对话框模板，并定位产生一个实际的标准按钮，其大小无关紧要，只需定位其位置及设置按钮的标题（本例为 OK、CANCEL）及按钮 ID（本例为 IDOK、IDCANCEL）；另外一定要在按钮的 Style 属性标签页中选中 Owner draw 复选项。

其次在资源 RC 文件中分别创建出四种状态（压下、弹起、有输入焦点和失效）的按钮位图形，其 ID 名为原标题文字串加字母 D、U、F、X 之一（如本例中原标准按钮标题为 OK，则位图 ID 分别为“OKD”、“OKU”、“OKF”、“OKX”，一定要用双引号！）。

再在对话框类定义中，分别加入内嵌的位图按钮成员数据（见前文 2 节），并利用 ClassWizard 设计其通告消息响应函数（类同于标准的按钮操作）。

最后在派生的对话框类 OnInitDialog() 成员函数内调用 CBitmapButton 类中的成员函数 AutoLoad() 装入自绘按钮位图，从而实现替换标准按钮。

```
BOOL CAboutDlg::OnInitDialog()
{ CDialog::OnInitDialog();
  ButtonOK.AutoLoad(IDOK, this); //装入自绘按钮位图
  ButtonCancel.AutoLoad(IDCANCEL, this);
  m-icon.SubclassDlgItem(IDC-BIGICON, this); //“动态子类化”
  m-icon.SizeToContent(); //置为位图大小
  CBrush BrushCyan(RGB(0,255,255)); //初始化涂刷为青色
  m-hCyanBrush = (HBRUSH)BrushCyan.GetSafeHandle();
  CBrush BrushRed(RGB(255,0,0)); //初始化涂刷为红色
  m-hRedBrush = (HBRUSH)BrushRed.GetSafeHandle();
  return TRUE;
}
```

2. 派生法内嵌位图按钮

首先也采用 3.1 节中第一步方法，设计对话框模板及定位标准按钮，并在 Style 属性标签页中选中 Owner draw 复选项（本例的按钮 ID 为 IDC-BIGICON），但不需加标题；

其次利用 ClassWizard 创建出一个 CButton 类的派生类（本例为 CDrawButton），并设计两个消息响应函数 DrawItem() 和 WM-ERASEBKGN 消息响应函数 OnEraseBkgnd() 及一个成员函数 SizeToContent()。

```
void CDrawButton::SizeToContent()
{ int cxIcon = ::GetSystemMetrics(SM-CXICON);//获得系统
  int cyIcon = ::GetSystemMetrics(SM-CYICON);//图标大小
  SetWindowPos(NULL, 0, 0, cxIcon * 2 + 8, cyIcon * 2 + 8,
    SWP-NOACTIVATE | SWP-NOMOVE | SWP-NOZORDER);
}
BOOL CDrawButton::OnEraseBkgnd(CDC * pDC)
{ return CButton::OnEraseBkgnd(pDC);
}
```

再在对话框类定义中内嵌此位图按钮对象成员数据变量（见前文 2 节），覆盖对话框类的 OnInitDialog() 成员函数，在其内将自绘按钮对象“动态子类化”（见前文 3.1 节）。

最后利用 ClassWizard 对此自绘按钮设计其单击右键通告消息响应函数(本例的实现是用户单击它,将在编辑条控制中显示"Click Telephone"文字串)。

```
void CAboutDlg::OnBigicon()
{ m-EditBox.SetWindowText("Click Telephone");
}
```

四、定制位图菜单

标准的 Windows 程序菜单为文字菜单,为使菜单直观形象,也可用位图形表示,位图可应用于菜单条的菜单项中,也可应用于浮动弹出式菜单项中(图 2 为在视窗口内单击右键而弹出的浮动菜单,包括有"画圆"和"画矩形"两子菜单项)。

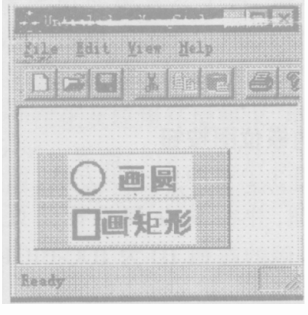

图 2 弹出式位图菜单

1. 使用位图菜单

在应用程序中使用位图菜单的一般方法为:首先创建位图 CBitmap 类对象;其次利用此类的成员函数 LoadBitmap()加载位图资源以初始化位图对象;最后将位图加到菜单项中(用 AppendMenu()函数)或控制修改某一菜单项为位图菜单项(用 ModifyMenu()函数),程序代码见 4.2 节。

2. 在浮动弹出式菜单中使用位图菜单

在应用程序的视窗类中响应单击右键消息,利用 ClassWizard 设计其消息响应函数,然后显示出浮动弹出式位图菜单项。

```
void CYangCtrlView::OnRButtonDown(UINT nFlags, CPoint point)
{ CBitmap CircleBitmap, RectBitmap;
  if(CircleBitmap.LoadBitmap(IDB-MENUCIRCLE)&&
  RectBitmap.LoadBitmap(IDB-MENURECT))
  { CMenu PopUpMenu;
    PopUpMenu.CreatePopupMenu(); //创建弹出式菜单
    PopUpMenu.AppendMenu(MF-BITMAP, ID-EDIT-UNDO,
      (const CBitmap *)&CircleBitmap);
    PopUpMenu.AppendMenu(MF-BITMAP, ID-EDIT-CUT,
      (const CBitmap *)&RectBitmap);
    ClientToScreen(&point); //转换坐标
    PopUpMenu.TrackPopupMenu(TPM-CENTERALIGN, point.x, point.y, this); } //显示出弹出式菜单
  else
  CView::OnRButtonDown(nFlags, point);
}
```

参考文献

[1] 何晓刚 译, Microsoft Visual C++ 4.0 教程, 1997, 北京:科学出版社

(来稿时间:1998 年 3 月)