单文档窗口分割视图和普通视图的切换

赵田雨[1] 王崇宇[2]
(1.上海理工大学机械工程学院 中国 上海 20093;2.即墨市技工学校 山东 青岛 266200)

摘要:本文介绍了利用 VISUAL C++6.0 编制含有分割视图和普通视图的基于单文档的多视图界面,以及视图之间的视图切换的方法。实现了在一个单文档框架中包含分割视图和普通视图,视图切换方便简单,满足了在一个程序中同时用分割视图和普通视图来显示数据,并用一个实例来详细说明编制的过程。

关键词:单文档;分割视图;普通视图;视图类;切换

The SDI split view and normal view switch

Abstract: This paper introduce programming a multi-view interface including split views and normal views based on the SDI by Visual C++ 6.0. It also gives a way of switching views. This method realizes in a SDI frame including split views and normal views, the switch is easy and convenient, satisfies the request of displaying data using the split view and normal view at the same time in a program. And then uses an example to explain the programming process.

Key words: SDI; split view; normal view; the view class; switch

1.引言

在编制的应用程序中,用户界面是用户和系统之间最直接的交互部分,我们利用 VC++来编制应用程序的可视化界面的时候,通常用的 MFC 框架, 即基于文档视图的应用程序。它提供了 2 种文档视图结构,一种是 SDI 结构,即单文档视图结构,二是 MDI 结构,即多文档多视图结构。但是应用 MFC 创建的框架都只有单一类的视图和功能,在实际应用过程中,它不能满足用户的需要,有时需要用户在不同的视图中进行操作,在同一个程序中包含多个视图也是经常遇到的。在我们做的一个项目中,用到了多视图的情况,包含分割视图和多个基于不同视图类的普通视图,在这里就介绍一下我们在面对这种情况的时候所用的方法。用这种方法实现在一个单文档多视图结构中包含分割视图,普通视图的操作界面,以及视图之间的切换。

2.原理概述

在建立一个基于单文档的工程之后,它会自动创建五个类,一个主框架窗口类(CMainFrame),一个以"C+工程名+App"为名字的类,它是主要应用程序类。一个以"C+工程名+Doc"为名字的类,它是文档类。一个以"C+工程名+View"为名字的类,它是视图类。还有一个 CAboutDlg 类,它是帮助对话框。单文档界面只有一个文档类,但可以有多个视图类。在默认建立的工程中只有一个视图类,因此视图是单一的视图。为了要实现多视图,我们就需要在工程中添加新的基于不同基类的视图类,每个和文档类关联的视图都有一个 control ID,这个 ID 是一个整数。如果只显示一个视图类, 这个类的 control ID 是 AFX_IDW_PANE_FIRST, 在多视图切换函数中切换的时候只要把要显示的视图的 control ID 设置为 AFX_IDW_PANE_FIRST,然后隐藏其他视图就可以实现窗口的切换,实现还是比较方便的。

切分窗口的实现要复杂一点,单文档只有一个主框架类,如果要分割窗口,就要切分主框架,使用一般的动态切分和静态切分切分完后,主框架窗口就被分割了,显示的时候就只能显示分割窗口,与普通窗口的切换变得很复杂,并且容易出错。我们提出一种新的方法,在类中添加一个新的框架,分割这个框架,然后把它作为一个视图,参与视图切换,这样就使得分割视图和普通视图的切换变得容易和方便。

窗口分割我们就要用到 CSplitterWnd 类,CSplitterWnd 像是一种特殊的框架窗口,每个窗口都被相同的或者不同的视图所填充。当窗口被切分后用户还可以使用鼠标移动切分条来调整窗口的相对尺寸。CSplitterWnd 有两种不同的切割窗口的方式。

一是动态切分,用到函数 Create(),函数声明如下:

BOOL Create (CWnd* pParentWnd, int nMaxRows, int nMaxCols, SIZE sizeMin, CCreateContext* pContext, DWORD dwStyle, UINT nID);

功能描述:该函数用来创建动态切分窗口。

参数含义:pParentWnd 指向父窗口的指针。

nMaxRows, nMaxCols 分割窗口的最大行数和列数。

sizeMin 子窗口的最小尺寸。

pContext 创建分割窗口的上下文。

dwStyle 分割视图的属性。

nID 子窗口的 ID 号。

动态切分可以让使用者通过拖曳分裂方块的使用, 将窗口切分。但是,动态切分最多只可以将窗口分裂为 2×2 个子视窗,不能进行混合分裂视窗,所有子窗口的属性和父窗口都是一样的,而且子窗口的数据通常来源于同一处。

另一种是静态切分,用到函数 CreateStatic(),函数声明如下:

BOOL CreateStatic (CWnd* pParentWnd,int nRows,int nCols, DWORD dwStyle,UINT nID)

功能描述:用来创建切分窗口。

参数含义:pParentWnd 切分窗口的父框架窗口。

nRows, nCols 窗口切分的列数和行数。

dwStyle 窗口的样式。

nID 子窗口的 ID 号。

静态切分,使用者除了可以调整子窗口的大小和进行混合分裂窗口外,最多可将窗口分裂为 16×16 个子窗口,每个子视窗可以有各自不同的视图类,各个子视窗显示的数据可以来自于不同的数据源。

窗口分割完后要用视图填充窗口, 这要用到函数 CreateView(),函数声明如下:

BOOL CreateView (int row, int col, CruntimeClass* pViewClass, SIZE sizeInit, CCreateContext* pContext);

功能描述:为静态切分的窗口的网格填充视图。在将视图于切分窗口联系在一起的时候必 须先将切分窗口创建好。

参数含义:row, col 将新视图与指定切分窗口所在的行和列的窗格联系起来

pViewClass 指定新视图的 CRuntimeClass 类

sizeInit 指定新视图开始的大小

pContext 创建视图的上下文构造指针

3.具体的步骤。

(1) 在主程序中添加一个框架类, 在视窗框架类中定义 CSplitterWnd 类的指针,来调用 CSplitterWnd 类。然后再添加其余的所需要的视图。

(2)重载视窗框架类中的 OnCreateClient()函数,单文档程序中是(CFrameWnd::OnCreateClient),分割新添加的框架类,建立静态分裂子视窗,为静态分裂子视窗填充视图。

(3)编写切换函数,建立维持各子视窗同步更新的机制。

(4)编程实例。

在这个实例中,有一个框架类,还有两个视图类,通过点击菜单栏中的菜单项,在他们之间进行切换,具体的步骤如下:

1. 首先应用 MFC AppWizard (exe) 新建一个基于单文档的名为 Yanshi 的工程,选项都使用默认即可。

2.在工程中新建一个框架类,CSplittedFrame 派生自 CFrameWnd,这个类的构造函数默认是 protected,为了能够调用,要改成 public。在头文件 SplittedFrame.h 中修改如下,

```
class CSplittedFrame : public CFrameWnd
{
  DECLARE_DYNCREATE(CSplittedFrame)
```

```
public:
  CSplittedFrame();
  virtual ~CSplittedFrame();
  protected:
     DECLARE_MESSAGE_MAP()
};
```

3.建立新的视图类。

在工程中新建一个 CView1 派生自 CView 类，在头文件 View1.h 中将构造函数由 protected 改为 public。然后在 CView1 的 OnDraw (CDC* pDC)函数中添加如下语句：

```
void CView1::OnDraw(CDC* pDC)
{
    CDocument* pDoc = GetDocument();
    pDC->TextOut(50,50,"这里用来显示文字");
}
```

接着新建一个 CFView1 派生自 CFormView 类，建立方法是先点击菜单栏的“Insert”点击“Resource”在弹出的对话框中选择“Dialog”->“IDD_FORMVIEW”，点击“New”，就插入了一个新的对话框，在对话框中拖入一个静态文本控件，属性标题改为“这里使用控件”，然后在属性里面选择“扩展样式”选择“静态边缘”。完成后点击“view”菜单中的“ClassWizard”弹出一个对话框询问是否新建一个类，选择新建一个类，然后写上类名和基类就新建了一个 CFormView 类。同样在头文件 FView1.h 中把构造函数由 protected 改为 public。

上面两个新建的类用来和 CSplittedFrame 类进行切换，下面的类用来填充在 CSplittedFrame 类中切分出的子窗口，新建一个 CTView 类，派生自 CTreeView，并且在头文件 StdAfx.h 中加入 #include < afxcview.h>，修改头文件 TView.h，把构造函数由 protected 改为 public。然后新建一个 CLView 类派生自 ClistView，修改头文件 LView.h，把构造函数由 protected 改为 public。

4.添加类指针。

在 CMainFrame 里加入头文件 #include "SplittedFrame.h" 和 #include “View1.h” 和 #include “FView1.h”。添加 CsplittedFrame 的指针，CView1 和 CFView1 的指针，还有一个整形变量，如下：

```
class CMainFrame : public CFrameWnd
{  ……
    protected:
    CSplittedFrame*      m_pSFrame;//框架类指针
    CView*               m_pView;//视图 CView1 指针
    CView*       m_pFView;//视图 CFormView1 指针
    int     m_CurrentID;//整形变量
    ……
};
```

5. 重写 CMainFrame 的 OnCreateClient()函数。

```
BOOL CMainFrame::OnCreateClient (LPCREATESTRUCT lpcs, CCreateContext* pContext)
{
    m_pView = new CView1;
    m_pView->Create(NULL, NULL, AFX_WS_DEFAULT_VIEW & ~WS_BORDER, CFrameWnd::rectDefault,this,NULL,pContext);
    m_pView->ShowWindow(SW_SHOW);
     m_pView->SetDlgCtrlID (AFX_IDW_PANE_FIRST);//设置视图 CFormView1 的 ID 为 AFX_IDW_PANE_FIRST
    pContext->m_pNewViewClass = (CRuntimeClass*)m_pView;//设置视图 CFormView1 为显示时默认的视图

    m_pFView = new CFView1;
    m_pFView->Create(NULL, NULL, AFX_WS_DEFAULT_VIEW & ~WS_BORDER, CFrameWnd::rectDefault,this,IDD_FORMVIEW, pContext);
    m_pFView->ShowWindow(SW_HIDE);

    m_pSFrame = new CSplittedFrame;
    m_pSFrame->Create (NULL, NULL, AFX_WS_DEFAULT_VIEW & ~WS_BORDER, CFrameWnd::rectDefault,this,NULL,0,pContext);
m_pSFrame->ShowWindow(SW_HIDE);
     return TRUE;
}
```

在 CMainFrame 中重载 OnDestroy()函数。

```
void CMainFrame::OnDestroy()
{   CFrameWnd::OnDestroy();
    delete m_pSFrame;
    delete m_pView;
    delete m_pFView;
}
```

6.切分窗口

在 CsplittecFrame 的头文件声明 CsplitterWnd 成员。

```
class CSplittedFrame : public CFrameWnd
{   ……
  protected:
     CSplitterWnd    m_wndSplitter;//声明框架成员
     ……
};
```

并在 SplittedFrame.cpp 添加头文件 #include “TView.h” 和 #include “LView.h”

重载 CSplittecFrame 的 OnCreateClient()函数

```
BOOL CSplittedFrame::OnCreateClient (LPCREATESTRUCT lpcs, CCreateContext* pContext)
{ m_wndSplitter.CreateStatic(this,1,2);
   m_wndSplitter.CreateView (0,0,RUNTIME_CLASS (CTView),CSize (200,0),pContext);
   m_wndSplitter.CreateView (0,1,RUNTIME_CLASS (CLView),CSize (0,0),pContext);
     return TRUE;
}
```

7.函数切换

在 CMainFrame 了添加一个函数 Switch(int nID)，这是一个布尔返回值的切换，通过判断返回值可以避免同一个视图被多次创建，这样保证只有一个视图被创建。

```
BOOL CMainFrame::Switch(int nID)
{
  if (nID == m_CurrentID)
  {
     return false;//如果视图已经创建，不在创建视图
  }
   switch(nID)
   {
        case IDD_FORMVIEW:
        m_pFView->ShowWindow(SW_SHOW);//显示视图
        m_pFView->SetDlgCtrlID(AFX_IDW_PANE_FIRST);

         m_pView->ShowWindow(SW_HIDE);//隐藏旧视图
         m_pView->SetDlgCtrlID(AFX_IDW_PANE_FIRST+2);

          m_pSFrame->ShowWindow(SW_HIDE);
          m_pSFrame->SetDlgCtrlID(AFX_IDW_PANE_FIRST+1);

          m_CurrentID = IDD_FORMVIEW;//保存当前视图 ID
          break;
          case AFX_IDW_PANE_FIRST+1:
          m_pSFrame->ShowWindow(SW_SHOW);
          m_pSFrame->SetDlgCtrlID(AFX_IDW_PANE_FIRST);

           m_pFView->ShowWindow(SW_HIDE);
           m_pFView->SetDlgCtrlID(IDD_FORMVIEW);

            m_pView->ShowWindow(SW_HIDE);
            m_pView->SetDlgCtrlID(AFX_IDW_PANE_FIRST+2);
```

```
                m_CurrentID = AFX_IDW_PANE_FIRST+1;
                break;
                case AFX_IDW_PANE_FIRST+2:
                    m_pView->ShowWindow(SW_SHOW);
                    m_pView->SetDlgCtrlID(AFX_IDW_PANE_FIRST);

                    m_pFView->ShowWindow(SW_HIDE);
                    m_pFView->SetDlgCtrlID(IDD_FORMVIEW);

                    m_pSFrame->ShowWindow(SW_HIDE);
                     m_pSFrame  ->SetDlgCtrlID  (AFX_IDW_PANE_
FIRST+1);
                     m_CurrentID = AFX_IDW_PANE_FIRST+2;
                     break;
            }
              RecalcLayout();
              return true;
        }
```

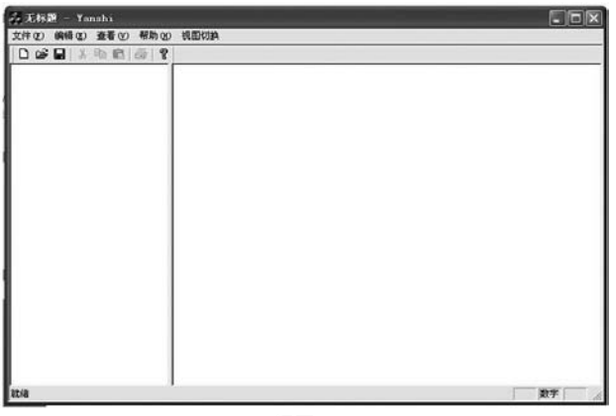
8.在菜单资源中加一个菜单,标题写“视图切换”。

然后添加子菜单项,“文本视图”,ID 为 IDD_WENBEN, 添加子菜单项“控件视图”,ID 为 IDD_KONGJIAN,添加子菜单项“切分窗口”,ID 为 IDD_QIEDEN。然后在 CMainFrame 里面响应菜单消息,添加函数如下:

```
void CMainFrame::OnKongjian()
{
        Switch (IDD_FORMVIEW);
}
void CMainFrame::OnWenben()
{
        Switch (AFX_IDW_PANE_FIRST+2);
}
void CMainFrame::OnQiefen()
{
       Switch (AFX_IDW_PANE_FIRST+1);
}
```

至此我们建立的程序实例现在就完成了,编译运行,程序即可运行,点击视图切换菜单的各个子菜单项,就可以进行视图的切换了。

视图 1



视图 2

9.结论

我们通过这个方法做出的多视图程序, 在使用和切换中都没有问题。因为 VC++的功能强大,用它来做程序界面的时候可以用很多方法,并能做出各种各样满足我们需要的程序界面,视图切换的方法也很多。相比较而言,我们的方法是其中一个比较简单易行的方法,不需要复杂的编程,就可以实现所需功能,用多种视图表现数据在程序使用中变得也是越来越普遍。我们的方法也有很多可以改进的地方,比如我们现在只是用到了 3 个视图的切换, 如果有更多的视图切换,只需要再添加新的视图,并改写切换函数就能实现,默认的显示视图也可以根据需要自己设定,有着很大的扩展性。

参考文献

[1]孙雄勇, Visual C++6.0 实用教程[M], 中国铁道出版社, 2004
[2]丁有和, Visual C++文档视窗设计[M], 青岛出版社, 2000
[3]郭庆民, Visual C++高级界面特效制作百例[M], 中国电力出版社, 2000
[4]Charles Petzold, Programming Windows (Fifth Edition)[M], Microsoft Press, 2004
[5]候俊杰, 深入浅出 MFC[M], 华中科技大学出版社, 2001.

(上接第 74 页)**管理人员(编号 char(15),姓名 char(15),密码 char(15))**

高级管理人员(编号 char(15),姓名 char(15),密码 char(15))

IP 地址表(IPchar(15),机房号 char(2),在线标志字节,收费标志 char(1))

上机记录(学号 char(15),IPchar(15),开始时间日期/时间,结束时间日期/时间)

上机历史记录(学号 char(15),IPchar(15),开始时间日期/时间,结束时间日期/时间,上机时间 int)

售票记录(售票员编号 char(2),售票员姓名 char(10),学生学号 char(15),金额 int,日期日期/时间)

4.结论

本文提出了计算机实验室管理系统的分析和设计思路,提出了计算机实验室信息化管理系统的设计思想,对计算机实验室信息化管理系统的软件结构和数据库结构进行了设计,详细介绍了系统各模块的功能。

参考文献

[1]卢显,王宇,吴忠望,杨健康.基于网络控制理论概念的网络控制论系统与分析.计算机工程与科学,2004 年第 26 卷第 3 期.
[2]张为,郭荷清,方南辉.基于面向对象技术的原型文档自适应器的设计.计算机应用与软件,2003 年第 20 卷第 7 期.
[3]周德明,张丽,谢谦.面向对象原型开发方法的一种实现策略。计算机研究与发展。1996 年第 4 期.

阅读此文的还阅读了：

1. 化解学生在学习截交线相惯线画法中的困难
2. 把公式表达式打印出来
3. 找回失落的文本框
4. 北京卷
5. 基于SQL Server的企业劳保管理系统数据库的设计
6. 视图与投影
7. 面向对象数据库中的视图
8. GB／T17451—1998《技术制图 图样画法 视图》简介
9. 基于MapX平台的地理信息系统开发技术的研究
10. 从三个方向看
11. 基于OO4O和VC 6.0实现Oracle数据库操作
12. 地图出版系统的四视图体系结构设计
13. 面向CBSD的软件体系结构模型研究
14. XQuery视图机制
15. 装配图中的二维图形消隐算法
16. 基于MVC模式的Web OA系统设计与研究

17. MVC设计模式研究

18. 机械制图中的一题多解剖析

19. 对国家标准《机械制图》部分内容的探讨

20. 怎样方便地将E数据生成一文本文件

21. 从不同方向看的画图策略

22. 框架设计通用化方法研究

23. 三视图的读识

24. 如何设计合理高效的数据库

25. 视图事务经历的串行化调度分析

26. PKTS view:一种多媒体数据库视图

27. Visual Foxpro中视图的运用

28. Excel单元格中显示公式

29. 用好Word“预览”视图快速了解文档内容

30. 基于Web的物理实验信息化教学系统的设计

31. 基于VC的单文档窗口分割

32. MVC设计模式在J2EE平台上的研究与实现

33. 浅淡VFP中动态条件的视图设计

34. ANSI/ISO的SQL和Foxpro的SQL的差异

35. 让PowerPoint中的对象整齐排列

36. 视图的科学定义和分类

37. 投影与视图

38. 藏传因明学遣他部分思辨特征分析

39. 基于ASP.NET的Web网络应用程序开发的安全策略实践

40. 玩转“魔方”——Excel数据透视表和透视图初探

41. 单文档窗口分割视图和普通视图的切换

42. 如何提高VFP中数据表的安全性

43. 形体分析与造型设计

44. 游龙SiteView V6.2网络管理软件

45. 基于CodeIgniter的精品课程自助建站系统设计与实现

46. 多视图的STEP模型管理

47. 《相似》《视图与投影》《解直角三角形》知识训练

48. 集成环境下PDM系统版本管理研究

49. 如何提高VFP中数据表的安全性

50. 视觉与视图