“文档-视”学生信息系统实验报告

王明 161220124 2018.12.9

一、 需求分析

实验目标是基于 MFC“文档-视”框架构建一个学生信息管理系统。功能方面需要完成学生信息的保存和导入、对学生信息进行添加、修改、删除、排序等操作。在界面方面需要构建一个带有滚动条的显示界面。

二、 设计思路

基于 MFC 框架来做，构建 4 个类：CStudentAdminApp（继承 CWinAppEx）、CStudentAdminDoc（继承 CDocument）、CStudentAdminView（继承 CScrollView）、StuDialog（继承 CDialog）。其中 StuDialog 是自定义的为了操作学生信息而添加的对话框类。CStudentAdminApp 是最顶层的类，实例化一个应用程序，提供了对 Windows 应用程序的各部分进行组合和管理的功能，其中包括对主窗口和文档模板的管理以及实现消息循环等。CStudentAdminDoc、CStudentAdminView 体现了“文档-视”的设计模式，文档类负责记录和处理数据、视类负责显示文档的数据以及实现对文档数据进行操作时与用户的交互功能。文档类和视类是对应的关系，一个是数据的内部处理，一个是数据的外部表现。StuDialog 类是为了用户更好地与视类进行交互而建立的对话框类。

总结起来就是，CStudentAdminApp 负责初始化应用以及管理各个模块，StuDialog 负责接收用户的输入，CStudentAdminDoc 负责对用户输入的数据进行存储和处理，CStudentAdminView 负责数据的格式化显示。

三、 核心代码

（1） 学生信息的添加功能（修改功能与之相似）：

```
void CStudentAdminDoc::OnAdd()
{
    // TODO: 在此添加命令处理程序代码
    StuDialog dlg = new StuDialog();  //弹出对话框
    dlg.sno = "";
    dlg.name = "";
    dlg.sex = "";
    dlg.birthdate = "";
    dlg.birthplace = "";
    dlg.address = "";
    if (dlg.DoModal() == IDOK)  //判断是否是确定
    {
        Info info;  //存储用户输入
        info.sno = dlg.sno;
        info.name = dlg.name;
        info.sex = dlg.sex;
        info.birthdate = dlg.birthdate;
        info.birthplace = dlg.birthplace;
        info.address = dlg.address;
        infolist.push_back(info);  //存入信息表中
        current = infolist.size() - 1;
        SetModifiedFlag(true);
        UpdateAllViews(NULL);  //更新视图
    }
}
```

(2)　学生信息的删除功能:

```
void CStudentAdminDoc::OnDelete()
{
    // TODO: 在此添加命令处理程序代码
    infolist.erase(infolist.begin() + current);
    if (current == infolist.size())
        current--;
    SetModifiedFlag(true);
    UpdateAllViews(NULL);
}
```

(3)　学生信息的排序功能:

```
struct      //自定义比较函数
{
    bool operator()(Info v1, Info v2) const
    {
        int x1 = _ttoi(v1.sno);
        int x2 = _ttoi(v2.sno);
        return x1 < x2;
    }
} sno_rank;


void CStudentAdminDoc::OnRankSno()
{
    // TODO: 在此添加命令处理程序代码
    std::sort(infolist.begin(), infolist.end(), sno_rank);  //利用sort函数排序
    SetModifiedFlag(true);
    UpdateAllViews(NULL);  //更新视图
}
```

(4)　学生信息的保存、导入功能：

```
void CStudentAdminDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
        // TODO: 在此添加存储代码
        ar << infolist.size();   //存入学生信息的条数
        for (int i = 0; i < infolist.size(); i++)  //存入每一条学生的信息
        {
            ar << infolist[i].sno << infolist[i].name << infolist[i].sex << infolist[i].birthdate << infolis
        }
    }
    else
    {
        // TODO: 在此添加加载代码
        int len = 0;
        ar >> len;  //读取信息长度
        infolist.clear();
        current = -1;
        for (int i = 0; i < len; i++)   //将信息读入构建好的结构中
        {
            Info info;
            ar >> info.sno >> info.name >> info.sex >> info.birthdate >> info.birthplace >> info.address;
            infolist.push_back(info);
            current = i;
        }
    }
}
```

(5)　学生信息的显示功能：

```
// TODO: 在此处为本机数据添加绘制代码

CFont *old_font = (CFont *)pDC->SelectStockObject(ANSI_FIXED_FONT);  //设置新字体
for (int i = 0, num = pDoc->infolist.size(); i < num; i++)
{
    Info *st = new Info();
    pDoc->get_info(i, st);
    CString temp;       //将学生的信息存储为一个格式化的CString类型变量
    temp.Format(_T("%-10.10s%-10.10s%-10.10s%-10.10s%-20.20s%-40.40s"), st->sno, st->name, st->sex, st->birthd
    if (i != pDoc->current)   //设置背景色和字体颜色（区分选中的信息）
    {
        pDC->SetTextColor(RGB(220, 20, 60));
        pDC->SetBkColor(RGB(255, 255, 255));
    }
    else
    {
        pDC->SetBkColor(RGB(220, 20, 60));
        pDC->SetTextColor(RGB(0, 0, 0));
    }
    pDC->TextOut(0, i * 20, temp);  //输出字符串
}
pDC->SelectObject(old_font);  //将字体还原
```

(6)　滚动条功能：

```cpp
void CStudentAdminView::OnInitialUpdate()
{
    CScrollView::OnInitialUpdate();
    CStudentAdminDoc* pDoc = GetDocument();
    CSize sizeTotal, sizePage, sizeLine;
    sizeTotal.cx = 1000;
    sizeTotal.cy = pDoc->infolist.size() * 20;  //重新设置滚动页的大小
    RECT rect;
    GetClientRect(&rect); //获取视窗口的大小
    sizePage.cx = rect.right - rect.left - 8;
    sizePage.cy = rect.bottom - rect.top - 20;
    sizeLine.cx = 8;
    sizeLine.cy = 20;
    SetScrollSizes(MM_TEXT, sizeTotal, sizePage, sizeLine);

}
void CStudentAdminView::OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)
{
    //在文档大小变化的时候动态改变滚动条的范围
    CStudentAdminDoc* pDoc = GetDocument();
    CSize sizeTotal, sizePage, sizeLine;
    sizeTotal.cx = 1000;
    sizeTotal.cy = pDoc->infolist.size() * 20;
    RECT rect;
    GetClientRect(&rect); //获取视窗口的大小
    sizePage.cx = rect.right - rect.left - 8;
    sizePage.cy = rect.bottom - rect.top - 20;
    sizeLine.cx = 8;
    sizeLine.cy = 20;
    SetScrollSizes(MM_TEXT, sizeTotal, sizePage, sizeLine);  //根据视窗大小调整滚动条的范围
    Invalidate();
}

void CStudentAdminView::OnSize(UINT nType, int cx, int cy)
{
    //在视窗大小变化的时候动态改变滚动条的范围
    CScrollView::OnSize(nType, cx, cy);
    CStudentAdminDoc* pDoc = GetDocument();
    CSize sizeTotal, sizePage, sizeLine;  //得到视窗的真实大小
    sizeTotal.cx = 1000;
    sizeTotal.cy = pDoc->infolist.size() * 20;
    sizePage.cx = cx - 8;
    sizePage.cy = cy - 20;
    sizeLine.cx = 8;
    sizeLine.cy = 20;
    SetScrollSizes(MM_TEXT, sizeTotal, sizePage, sizeLine);
}
```

```
void CStudentAdminView::OnLButtonDown(UINT nFlags, CPoint point)
{
    // TODO: 在此添加消息处理程序代码和/或调用默认值

    //利用DC类来讲物理坐标转化为逻辑坐标
    CClientDC dc(this);
    OnPrepareDC(&dc, NULL);
    dc.DPtoLP(&point);
    CStudentAdminDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
    int y = point.y;
    int line = y / 20;
    if(line < pDoc->infolist.size())
        pDoc->current = line;
    Invalidate();

    CScrollView::OnLButtonDown(nFlags, point);
}
```

四、 实验感想

这次实验非常充分体现了 MFC“文档-视”的设计模式，该模式分工合理、与用户的交互良好，但是也正是有可能因为其框架构建得太全面，留给设计者的灵活度就相对较低，这可能是 MFC 逐渐淘汰的原因。