

第5章 对话框与控件

- 对话框与控件的概念
- 通用对话框类的用法
- 多种简单控件的例子
- 两种复杂控件的例子
- 带对话框的单文档应用程序

对话框的概念(1)

- 对话框(Dialog)是收集信息或提供反馈的窗口，它通过控件与用户交互
- 控件(Controls)是执行用户动作的窗口，通常由父窗口(对话框、边框窗口、视图或控件栏)所有
- 对话框可独自形成应用程序，或附属于单文档、多文档应用程序

对话框的概念 (2)

■ 消息对话框用于显示提示信息

- ✓ MessageBox(L"Overwrite existing file?", L"Message", MB_OK|MB_ICONQUESTION);

■ 消息对话框的布局

- ✓ MB_YESNO、MB_OKCANCEL MB_RETRYCANCEL、
MB_ABORTRETRYIGNORE、MB_ICONWARNING



■ 消息对话框的返回值

- ✓ IDYES、IDNO、IDOK、IDCANCEL、IDABORT、
IDIGNORE与IDRETRY

对话框的概念 (3)

■ 对话框应用程序设计流程

- ✓ 创建对话框应用程序框架
- ✓ 用对话框编辑器设计对话框
- ✓ 为对话框中的控件创建消息
- ✓ 为控件添加成员变量

对话框的概念(4)

■ 模态对话框

- ✓ 在对话框打开时，父窗口不能执行操作
- ✓ 用DoModal加载模板与显示对话框

■ 非模态对话框

- ✓ 在对话框打开时，父窗口能执行操作
- ✓ 自己调用构造函数，用Create加载模板，用ShowWindow显示对话框

对话框的概念 (5)

■ 模态对话框

```
CMyDialog dlg;  
dlg.DoModal();
```

■ 非模态对话框

```
CMyDialog *pDlg;  
pDlg=new CMyDialog(this);  
pDlg->Create(IDD_DIALOG1);  
pDlg->ShowWindow(SW_SHOW);
```

对话框的概念(6)

- 在程序中使用对话框的流程
 - ✓ 用对话框编辑器设计对话框
 - ✓ 用类向导创建对话框类
 - ✓ 为对话框中的控件创建消息
 - ✓ 为对话框中的控件添加成员变量
 - ✓ 创建并显示对话框

对话框的例子(1)

例5-1

■ 设计对话框控件

控件	控件标识	说明
Static Text	IDC_STATIC	学号
Static Text	IDC_STATIC	姓名
Static Text	IDC_STATIC	年龄
Edit Box	IDC_ID	
Edit Box	IDC_NAME	
Edit Box	IDC_AGE	
Button	IDC_TEST	Test

对话框的例子(2)

■ 控件添加成员变量

控件	控件标识	成员变量
Edit Box	IDC_ID	m_id
Edit Box	IDC_NAME	m_name
Edit Box	IDC_AGE	m_age

对话框的例子(3)

不要写在构造函数中

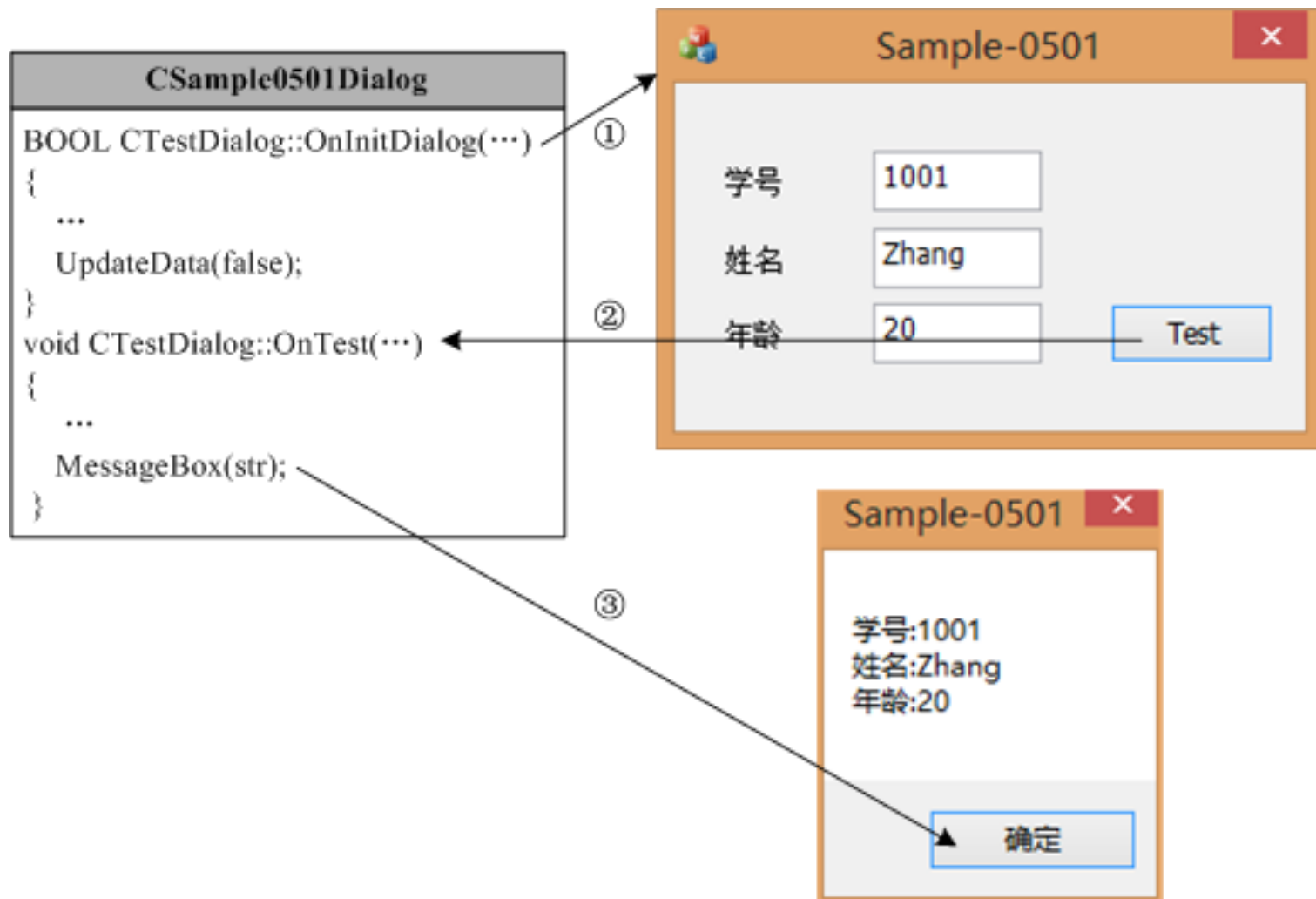
- 在CTestDialog::OnInitDialog() 中

```
m_id=L"1001";  
m_name=L"Zhang";  
m_age=20;  
UpdateData(false);
```

- 在CTestDialog::OnTest() 中

```
UpdateData(true);  
CString str;  
str.Format(L"学号:%s\n姓名:%s\n年龄:%d",  
m_id, m_name, m_age);  
MessageBox(str);
```

对话框的例子(4)



通用对话框类(1)

- 通用对话框 (CCommonDialog) 是预定义对话框，执行各种标准的操作

通用控件类	功能
CColorDialog	“颜色”对话框
CFileDialog	“文件”对话框
CFindReplaceDialog	“查找替换”对话框
CFontDialog	“字体”对话框
CPrintDialog	“打印”对话框
CPageSetupDialog	“页面设置”对话框

通用对话框类(2)

- CFontDialog是字体对话框类
- CFontDialog dlg;
- CFontDialog的成员函数
 - ✓ GetCurrentFont(): 获得字体样式
 - ✓ GetColor(): 获得字体颜色
 - ✓ IsBold(): 字体是否为粗体
 - ✓ IsItalic(): 字体是否为斜体
 - ✓ IsUnderline(): 字体是否带下划线

通用对话框类(3)

- CColorDialog是颜色对话框类
- CColorDialog dlg;
- CColorDialog dlg(RGB(255, 0, 0));
- CColorDialog的成员函数
 - ✓ GetColor(): 获得颜色
 - ✓ GetSavedCustomColors(): 获得自定义颜色
 - ✓ SetCurrentColor(): 设置颜色

通用对话框类(4)

- `CFileDialog`是文件对话框类
- `CFileDialog dlg(true);`
- `CFileDialog dlg(true, NULL, NULL, OFN_HIDEREADONLY, L"C++ Source|*.cpp");`
- `CFileDialog`的成员函数
 - ✓ `GetFileName()`: 获得文件名
 - ✓ `GetPathName()`: 获得文件路径
 - ✓ `GetFileExt()`: 获得文件扩展名

通用对话框类(5)

- `CPrintDialog`是打印对话框
- `CPrintDialog dlg(true);`
- `CPrintDialog`的成员函数
 - ✓ `GetDefaults()`: 获得默认打印机
 - ✓ `GetCopies()`: 获得打印份数
 - ✓ `GetFromPage()`: 获得打印起始页
 - ✓ `GetToPage()`: 获得打印终止页

通用对话框类(6)

- `CFindReplaceDialog`是查找对话框
- `CFindReplaceDialog* pDlg=new CFindReplaceDialog();`
- `pDlg->Create(true, NULL);`
- `pDlg->Create(false, NULL, NULL, FR_MATCHCASE);`
- `CFindReplaceDialog`的成员函数
 - ✓ `GetFindString()`: 获得查找的字符串
 - ✓ `GetReplaceString()`: 获得替换的字符串
 - ✓ `MatchWholeWord()`: 全字匹配状态

通用对话框的例子(1)

例5-2

- 在CTestView类中

```
LOGFONT m_font;  
COLORREF m_color;
```

- 在CTestView构造函数中

```
m_font.lfHeight=30;  
m_font.lfEscapement=0;  
m_font.lfWeight=FW_NORMAL;  
m_font.lfCharSet=GB2312_CHARSET;  
m_color=RGB(255, 0, 0);
```

通用对话框的例子(2)

- 在CTestView::OnDraw() 中

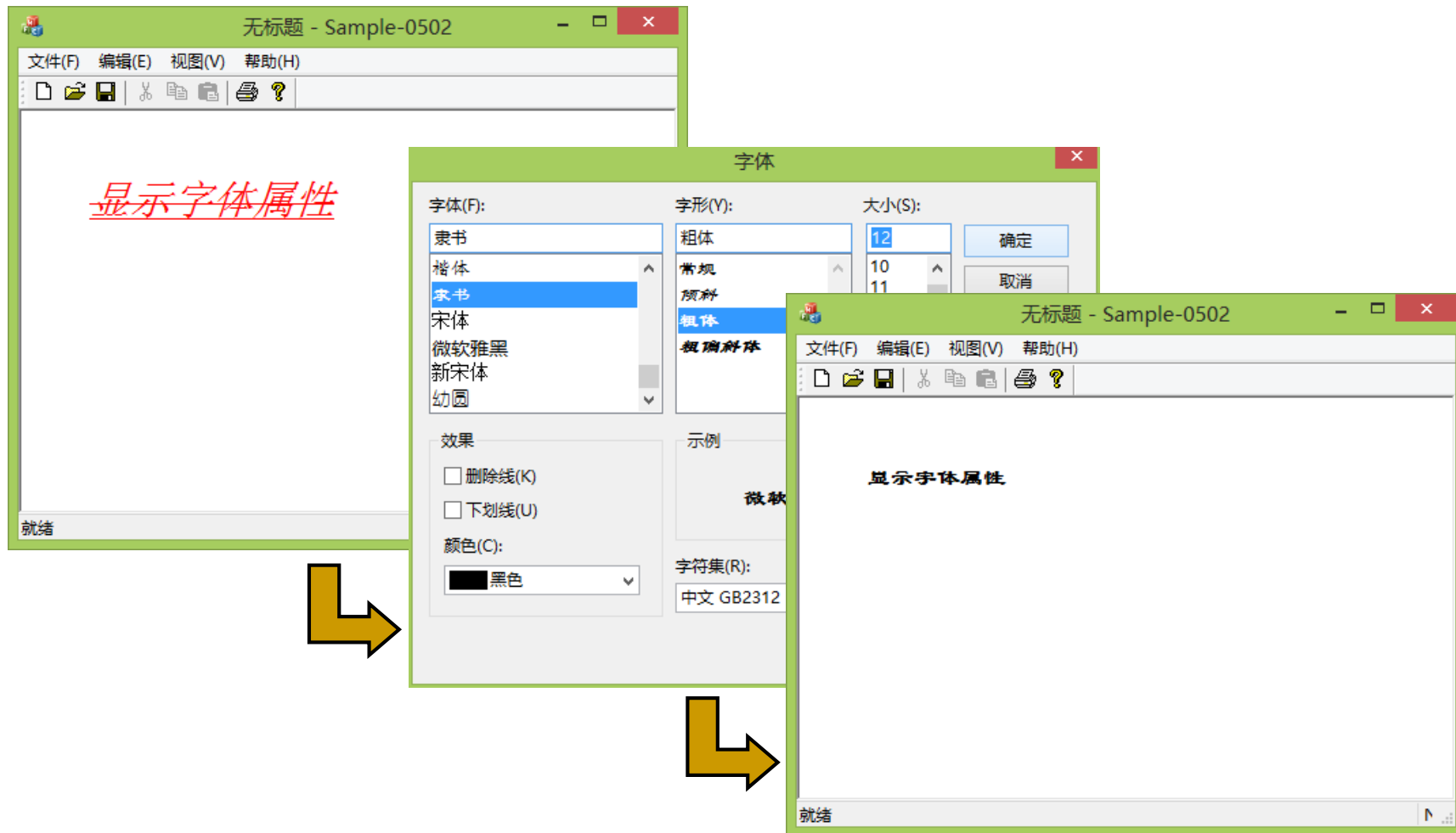
```
CFont newFont;  
newFont.CreateFontIndirect(&m_font);  
pDC->SelectObject(&newFont);  
pDC->SetTextColor(m_color);  
pDC->TextOutW(50, 50, L"显示字体属性");
```

通用对话框的例子(3)

- 在CTestView::OnDialog() 中

```
CFontDialog dlg;  
if (dlg.DoModal() == IDOK)  
{  
    dlg.GetCurrentFont(&m_font);  
    m_color = dlg.GetColor();  
    Invalidate(true);  
}
```

通用对话框的例子(4)



哪个通用对话框类支持非模态的对话框？

- ☐ A CFileDialog
- ☐ B CFontDialog
- ☐ C CPrintDialog
- ☒ D CFindReplaceDialog

提交

控件类型(1)

控件类	控件	说明
CStatic	静态控件	标识其它控件
CButton	按钮控件	按钮、单选钮与复选框
CEdit	编辑框控件	文本输入
CListBox	列表框控件	字符串列表
CComboBox	组合框控件	编辑框与列表框的结合
CScrollBar	滚动条控件	对话框中的滚动条
CAnimateCtrl	动画控件	显示AVI视频文件
CHeaderCtrl	标题控件	显示在文本列上的按钮

控件类型 (2)

控件类	控件	说明
CImageList	图像列表	图标组成的列表
CListCtrl	列表控件	图标与文字组成的列表
CProgressCtrl	进展条控件	任务完成进展
CRichEditCtrl	格式编辑框	多格式的文本输入
CSliderCtrl	滑动条控件	包含滑动条与刻度
CSpinButtonCtrl	旋转框控件	通过双向箭头增减值
CStatusBarCtrl	状态栏控件	状态信息显示栏
CToolBarCtrl	工具栏控件	工具信息相示栏

控件类型 (3)

控件类	控件	说明
CTabCtrl	标签控件	显示多页信息或控件
CTreeCtrl	树状列表控件	树状的层次列表结构
CBitmapButton	位图按钮控件	位图标签的按钮
CDragListBox	拖放列表控件	支持拖放的列表
CDateTimeCtrl	日期时间控件	显示日期与时间
CMonthCalCtrl	月历控件	显示月历
CIPAddressCtrl	IP地址控件	显示IP地址

静态控件(1)

- 静态控件(CStatic)用于显示文本，通常不进行输入与输出
- CStatic类用于控制静态控件，例如改变静态控件显示内容
 - ✓ IDC_STATIC到IDC_CTRLID、IDC_CTRLNAME、IDC_CTRLAGE

静态控件 (2)

例5-3

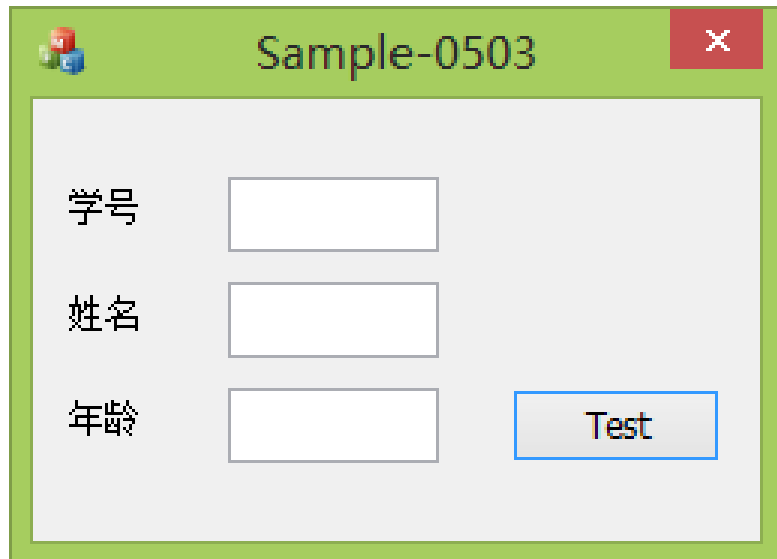
- 为静态控件添加成员变量

控件	控件标识	成员变量
Static Box	IDC_CTRLID	m_ctrlId
Static Box	IDC_CTRLNAME	m_ctrlName
Static Box	IDC_CTRLAGE	m_ctrlAge

- 在CTestDialog::OnTest() 中

```
m_ctrlId.SetWindowText(L"ID");  
m_ctrlName.SetWindowText(L"NAME");  
m_ctrlAge.SetWindowText(L"AGE");
```

静态控件 (3)



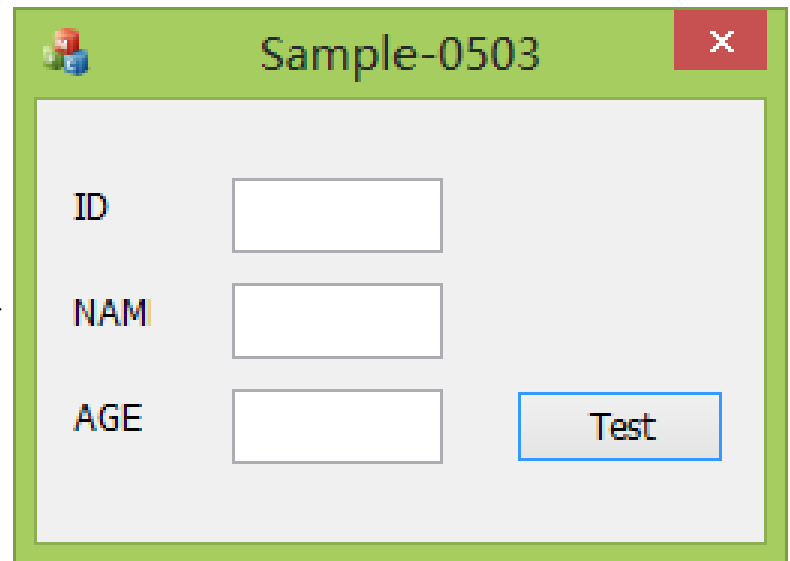
Sample-0503

学号

姓名

年龄

Test



Sample-0503

ID

NAM

AGE

Test

按钮控件(1)

- 按钮控件(CButton)是一种子窗口，通过单击执行某种操作
- 按钮控件类型：复选框、单选钮和下压按钮
- CButton类的成员函数
 - ✓ GetIcon() 与 SetIcon()
 - ✓ GetCheck() 与 SetCheck()
 - ✓ GetButtonStyle() 与 SetButtonStyle()

按钮控件 (2)

例5-4

- 修改按钮控件属性
 - ✓ 选中IDC_TEST的Icon属性
- 为按钮控件添加成员变量
 - ✓ IDC_TEST对应于m_ctrlTest
- 在CTestDialog::OnInitDialog() 中

```
HICON hIcon;  
hIcon=AfxGetApp()->LoadIcon(IDR_MAINFRAME);  
m_ctrlTest.SetIcon(hIcon);
```

按钮控件 (3)

- 增加一个复选框控件
 - ✓ IDC_CHECK属于按钮控件
- 在CTestDialog::OnCheck() 中

```
CButton *pCheck=(CButton*)GetDlgItem  
(IDC_CHECK);  
CString str;  
str.Format(L"复选框状态:%d", pCheck->  
GetCheck());  
MessageBox(str);
```

组合框控件(1)

- 列表框控件(CListBox)
 - ✓ 用于显示列表项，查看和选择列表项
- 编辑框控件(CEdit)
 - ✓ 用于输入文本信息
- 组合框控件(CComboBox)
 - ✓ 由列表框和编辑框控件组成

组合框控件 (2)

■ CComboBox类的成员函数

- ✓ GetCurSel: 获得列表当前项位置
- ✓ GetLBText: 获得列表指定项内容
- ✓ SetCurSel: 设置列表当前项位置
- ✓ AddString: 列表结尾添加字符串
- ✓ InsertString: 列表中插入字符串
- ✓ DeleteString: 列表中删除字符串

组合框控件(3)

例5-5

■ 设计对话框控件

控件	控件标识	说明
Static Text	IDC_STATIC	Number
Combo Box	IDC_NUMBER	
Group Box	IDC_DISPLAY	Display

组合框控件(4)

- 设置IDC_NUMBER属性
 - ✓ 添加数据(First、Second与Third)
 - ✓ 设置模式(DropDown与DropList)
- 为控件添加成员变量

控件	变量类型	成员变量
IDC_NUMBER	Value CString	m_number
IDC_NUMBER	Control	m_ctrlNumber
IDC_DISPLAY	Control	m_ctrlDisplay

组合框控件 (5)

- 在CTestDialog类定义中

```
private:  
    CRect rect;
```

- 在CTestDialog::OnInitDialog() 中

```
m_ctrlDisplay.GetWindowRect(&rect);  
ScreenToClient(&rect);  
m_ctrlNumber.SetCurSel(0);  
m_ctrlNumber.GetLBText(0, m_number);
```

GetLBText：获得列表指定项内容

组合框控件 (6)

选别的项

- 在CTestDialog::OnPaint()的else中

```
CPaintDC dc(this);  
dc.SetBkMode(TRANSPARENT);  
dc.SetTextColor(RGB(0, 0, 255));  
dc.TextOutW(rect.left+20, rect.top+20,  
m_number);
```

- 在CTestDialog::OnSelchangeNumber()中

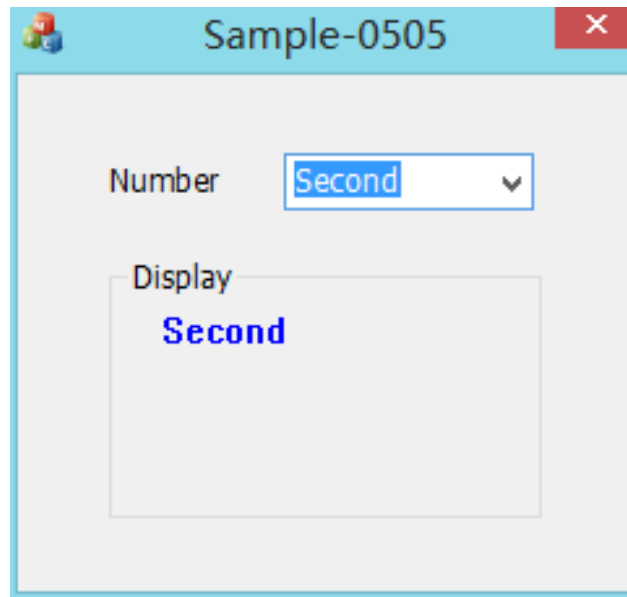
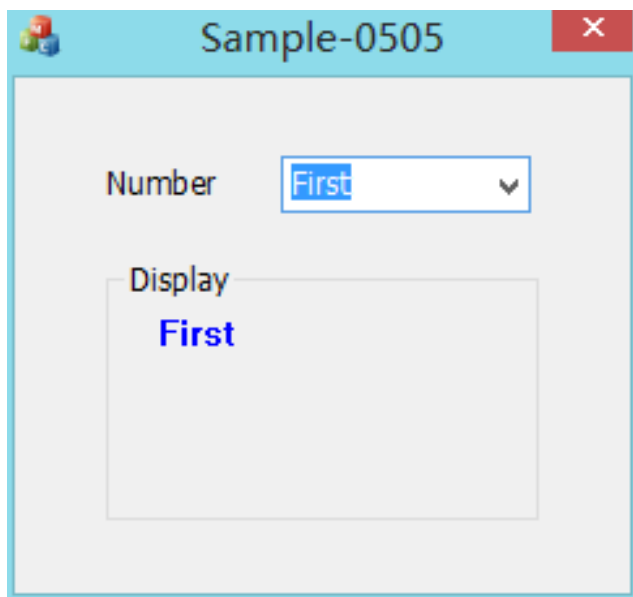
```
m_ctrlNumber.GetLBText(m_ctrlNumber.GetCu  
rSel(), m_number);  
InvalidateRect(&rect);
```

组合框控件(7)

编辑了框内的内容

- 在CTestDialog::OnEditchangeNumber() 中

```
m_ctrlNumber.GetWindowText(m_number);  
InvalidateRect(&rect);
```



标准模板库

39-75没有录屏回放；

■ STL (Standard Template Library)

- ✓ 称为标准模板类库，它是一种基于模板的容器类库
- ✓ 包括常用数据结构，例如链表、队列和栈
- ✓ 包括常用算法，例如排序和查找

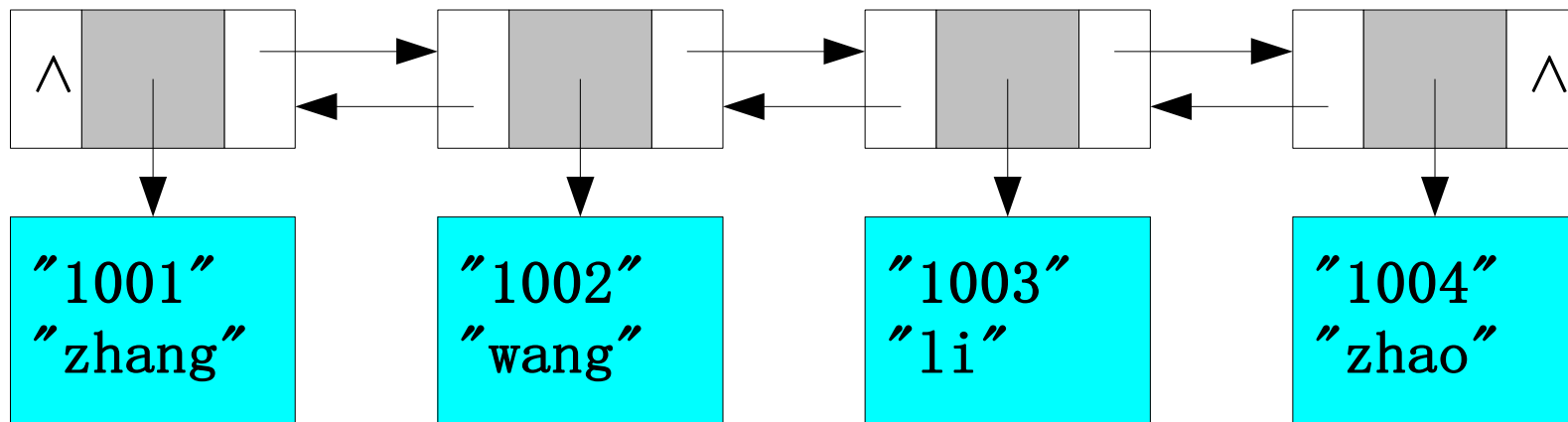
列表框控件(1)

■ 定义类模板

✓ CTypedPtrList <CObList, CStudent*>

m_pDataList

m_pDataList



列表框控件 (2)

■ m_pDataList的成员函数

- ✓ GetHeadPosition() 与 GetTailPosition()
- ✓ GetAt()、GetPrev() 与 GetNext()
- ✓ AddHead() 与 AddTail()
- ✓ InsertAfter() 与 InsertBefore()
- ✓ RemoveHead() 与 RemoveTail()
- ✓ RemoveAt() 与 RemoveAll()
- ✓ GetCount() 与 IsEmpty()

列表框控件 (3)

例5-6

■ 设计对话框控件

控件	控件标识	说明
Edit Box	IDC_ID	Id
Edit Box	IDC_NAME	Name
List Box	IDC_LIST	Student List
Button	IDC_PREVIOUS	Previous
Button	IDC_NEXT	Next
Button	IDC_ADD	Add
Button	IDC_CLOSE	Close

列表框控件(4)

- 添加CStudent类，基类为CObject

```
public:  
    CString m_id;  
    CString m_name;  
    CStudent(CString id, CString name)  
    { m_id=id;  
      m_name=name; };
```

列表框控件 (5)

■ 为控件添加成员变量

控件	变量类型	成员变量
IDC_ID	Value CString	m_id
IDC_NAME	Value CString	m_name
IDC_LIST	Control	m_ctrlList

■ 在CTestDialog类定义中

```
private:  
    POSITION m_pos;  
    CTypedPtrList <CObList, CStudent*>  
m_pDataList;
```

列表框控件 (6)

■ 在CTestDialog::OnAdd() 中

```
UpdateData(true);  
if(m_id.GetLength()==0)  
{ MessageBox(L"ID cannot empty!");  
  (CEdit*)GetDlgItem(IDC_ID)->SetFocus(); return; }  
if(m_name.GetLength()==0)  
{ MessageBox(L"Name cannot empty!");  
  (CEdit*)GetDlgItem(IDC_NAME)->SetFocus(); return; }  
CStudent *pStudent; //向链表追加结点  
pStudent=new CStudent(m_id,m_name);  
m_pDataList.AddTail(pStudent);  
m_pos=m_pDataList.GetTailPosition();  
CString str; //向ListBox追加字符串  
str=m_id+" "+m_name; m_ctrlList.AddString(str);  
m_ctrlList.SetCurSel(m_ctrlList.GetCount()-1);
```

列表框控件(7)

■ 在CTestDialog::OnPrevious()中

```
if(m_pos!=NULL)
{ //计算上一条记录在链表中位置
    if(m_pos==m_pDataList.GetHeadPosition())
        m_pos=m_pDataList.GetTailPosition();
    else m_pDataList.GetPrev(m_pos);
    //从链表中取出上一条记录
    CStudent *pStudent=m_pDataList.GetAt(m_pos);
    m_id=pStudent->m_id;  m_name=pStudent->m_name;
    UpdateData(false);
    //计算上一条记录在ListBox中位置并选中
    int pos=m_ctrlList.GetCurSel();
    if(pos==0)  pos=m_ctrlList.GetCount()-1;
    else  pos--;
    m_ctrlList.SetCurSel(pos); }
else  MessageBox(L"No Item!");
```

列表框控件 (8)

■ 在CTestDialog::OnNext() 中

```
if(m_pos!=NULL)
{ //计算下一条记录在链表中位置
    if(m_pos==m_pDataList.GetTailPosition())
        m_pos=m_pDataList.GetHeadPosition();
    else m_pDataList.GetNext(m_pos);
    //从链表中取出下一条记录
    CStudent *pStudent=m_pDataList.GetAt(m_pos);
    m_id=pStudent->m_id;  m_name=pStudent->m_name;
    UpdateData(false);
    //计算下一条记录在ListBox中位置并选中
    int pos=m_ctrlList.GetCurSel();
    if(pos==m_ctrlList.GetCount()-1)  pos=0;
    else  pos++;
    m_ctrlList.SetCurSel(pos); }
else  MessageBox(L"No Item!");
```

列表框控件 (9)

- 在CTestDialog::OnClose() 中

```
if (MessageBox (L"Close Dialog?", L"Close",  
MB_OKCANCEL | MB_ICONQUESTION) == IDOK)  
    CDialog::OnCancel();
```

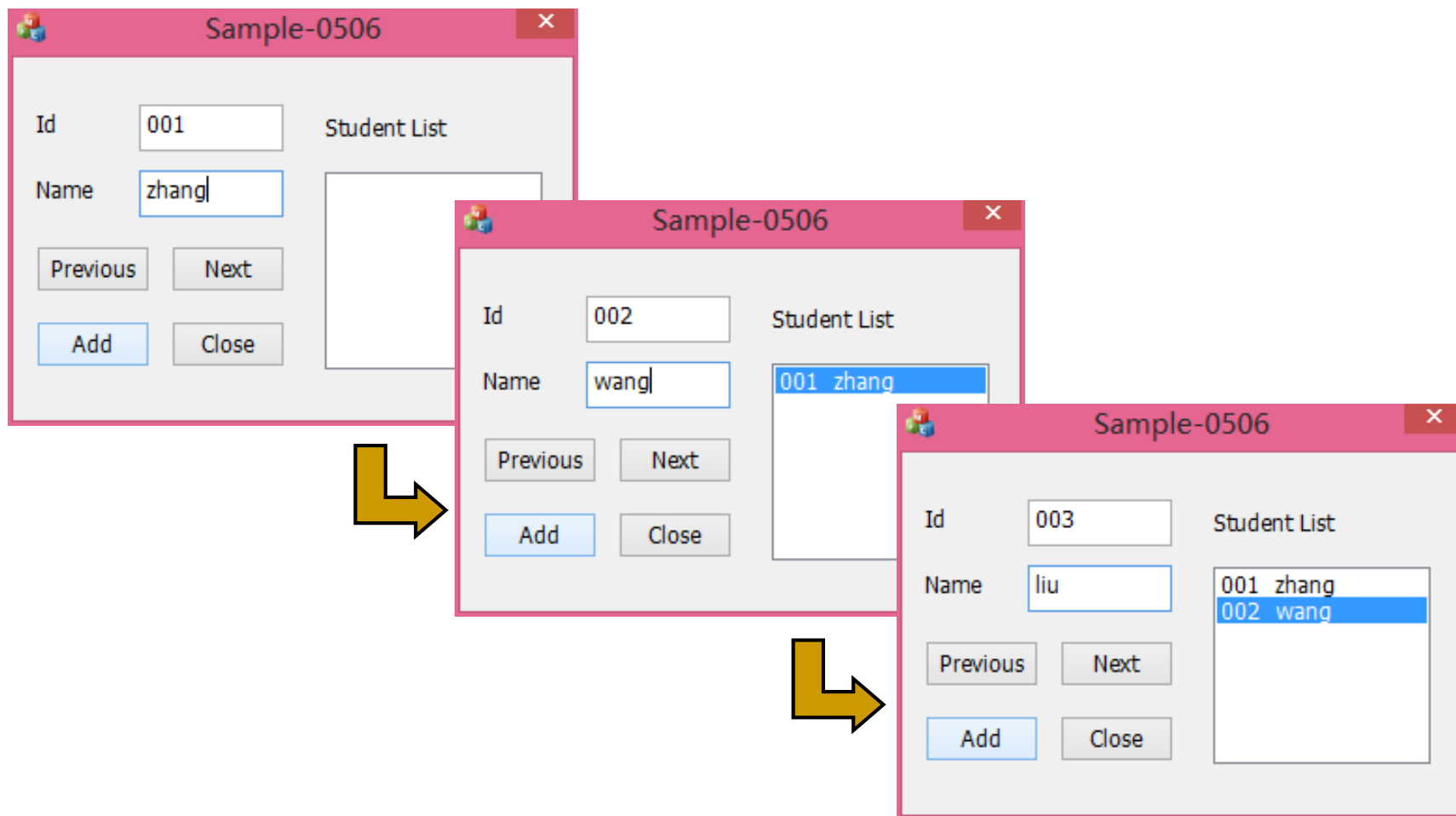
- 在CTestDialog::OnDestroy() 中

```
if (!m_pDataList.IsEmpty())  
    delete m_pDataList.RemoveHead();
```

- 在CTestDialog::OnInitDialog() 中

```
m_pos=NULL;
```


列表框控件 (10)



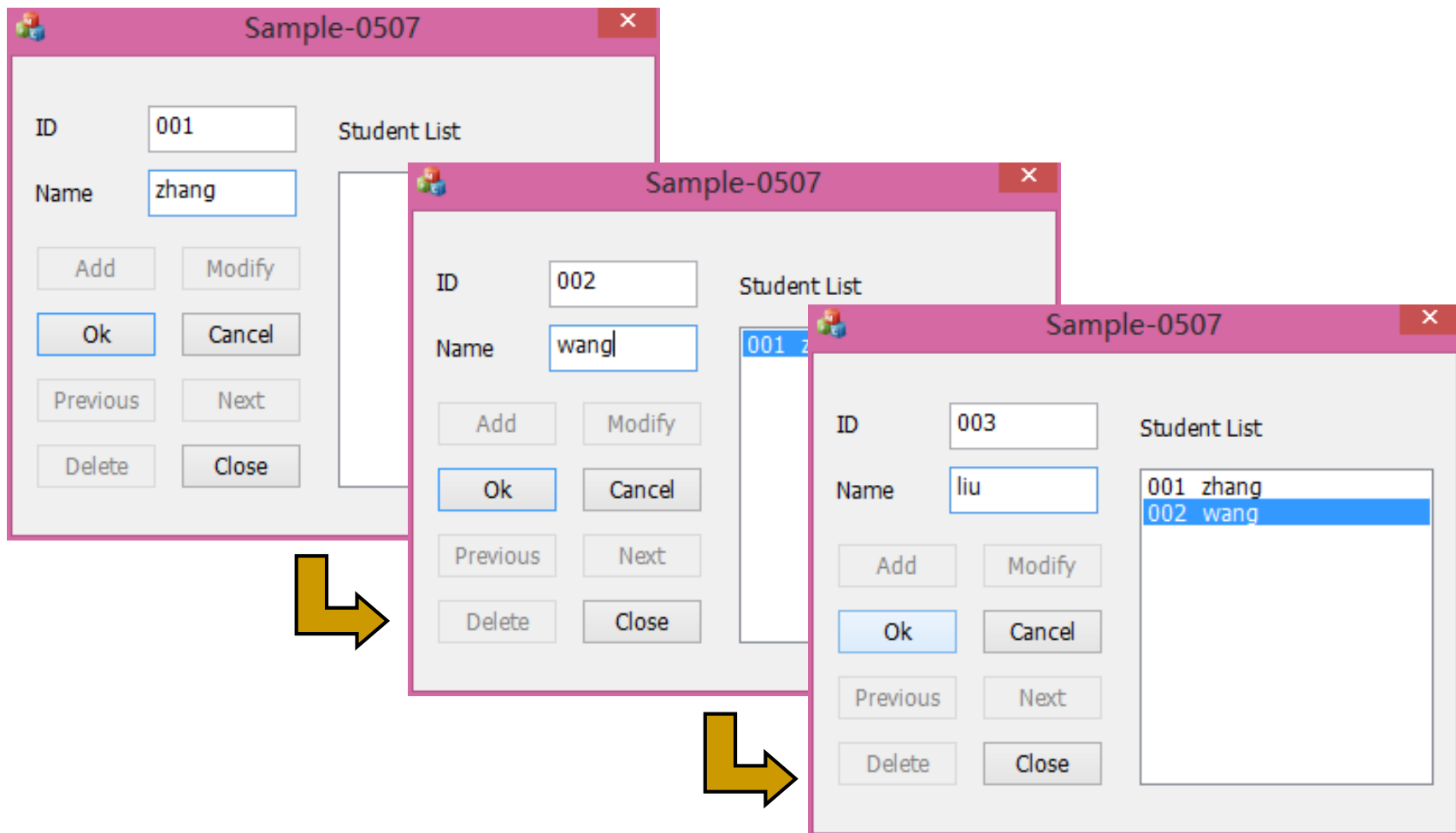
列表框控件(11)

例5-7

■ 通过状态图来控制按钮状态

```
enum STATE
{
    STARTSTATE,      //开始状态
    BROWSESTATE,     //浏览状态
    ADDSTATE,         //增加状态
    MODIFYSTATE,     //修改状态
    FINALSTATE,      //结束状态
};
```

列表框控件 (12)



哪个标识的控件不能设置成员变量？

- ☐ A IDC_BUTTON
- ☒ B IDC_STATIC
- ☐ C IDC_SLIDER
- ☐ D IDC_COMBOBOX

提交

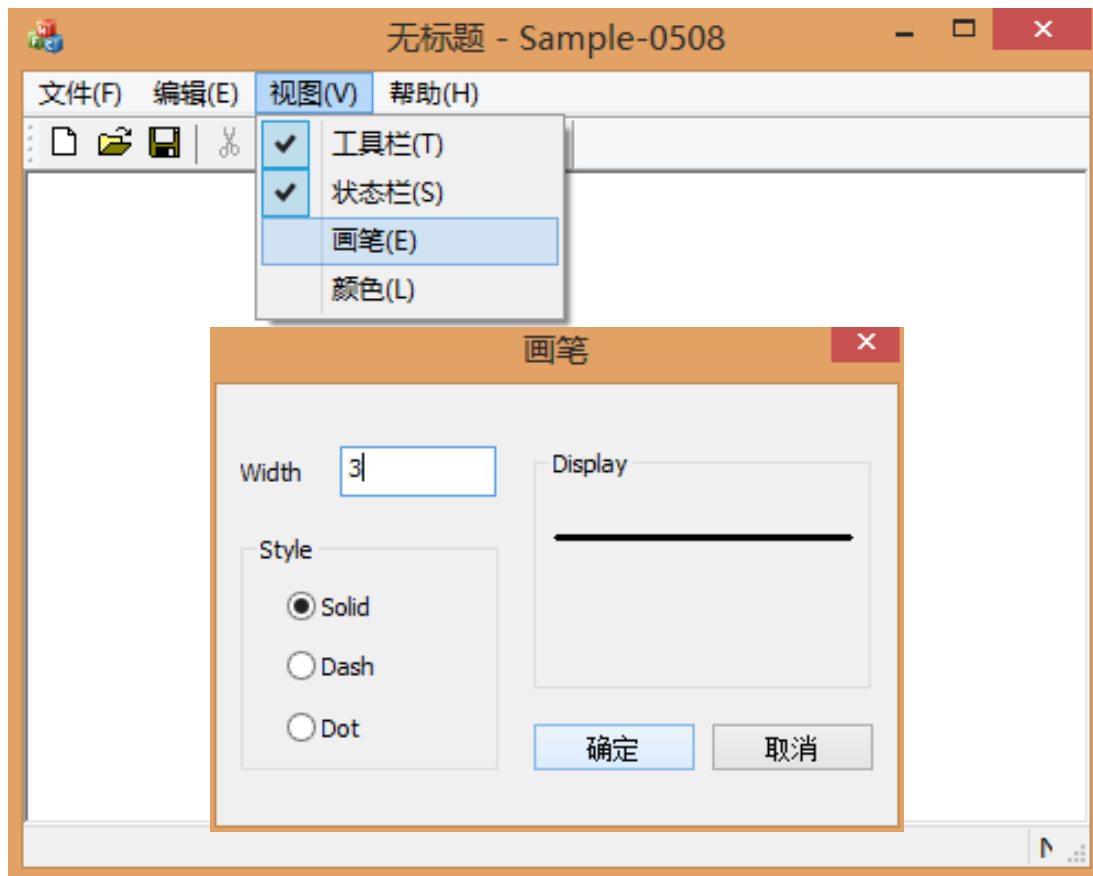
绘图程序的例子(1)

- 带对话框的单文档应用程序
 - ✓ 增加画笔对话框，改变画笔的样式与宽度
 - ✓ 增加颜色对话框，改变画笔的颜色
 - ✓ 增加菜单项与工具栏按钮，打开两种对话框
 - ✓ 通过画笔在视图中画一个矩形

绘图程序的例子(2)

例5-8

■ 用户界面要求



绘图程序的例子(3)

- 在菜单中增加菜单项
 - ✓ ID_VIEW_PEN 画笔(&E)
 - ✓ ID_VIEW_COLOR 颜色(&L)
- 在工具栏中增加工具栏按钮
 - ✓ ID_VIEW_PEN
 - ✓ ID_VIEW_COLOR
- 在快捷键列表中增加快捷键
 - ✓ ID_VIEW_PEN Ctrl+E
 - ✓ ID_VIEW_COLOR Ctrl+L

绘图程序的例子(4)

■ 设计对话框控件

控件	控件标识	说明
Static Text	IDC_STATIC	Width
Edit Box	IDC_WIDTH	
Group Box	IDC_STATIC	Style
Radio Button	IDC_SOLID	Solid
Radio Button	IDC_DASH	Dash
Radio Button	IDC_DOT	Dot
Group Box	IDC_DISPLAY	Display

绘图程序的例子(5)

■ 为控件添加成员变量

控件	变量类型	成员变量
IDC_WIDTH	Value int	m_width
IDC_SOLID	Value int	m_style
IDC_DISPLAY	Control	m_ctrlDisplay

■ 在CPenDialog类定义中

```
private:  
    CRect rect;
```

绘图程序的例子(6)

- 在CPenDialog::OnInitDialog() 中

```
m_ctrlDisplay.GetWindowRect(&rect);  
ScreenToClient(&rect);
```

- 在CPenDialog::OnChangeWidth() 中

```
UpdateData(true);  
if(m_width>1 && m_width<7)  
    InvalidateRect(&rect);
```

绘图程序的例子(7)

- 在CPenDialog::OnSolid() 中

```
m_style=0;  
InvalidateRect(&rect);
```

- 在CPenDialog::OnDash() 中

```
m_style=1;  
InvalidateRect(&rect);
```

- 在CPenDialog::OnDot() 中

```
m_style=2;  
InvalidateRect(&rect);
```

绘图程序的例子(8)

■ 在CPenDialog::OnPaint() 中

```
CPen newPen;  
switch(m_style)  
{  
case 0:  
    newPen.CreatePen(PS_SOLID, m_width, RGB(0, 0, 0));  
    break;  
case 1:  
    newPen.CreatePen(PS_DASH, m_width, RGB(0, 0, 0));  
    break;
```

绘图程序的例子(9)

■ 在CPenDialog::OnPaint() 中

```
case 2:
    newPen.CreatePen(PS_DOT, m_width, RGB(0, 0, 0));
    break;
default:
    newPen.CreatePen(PS_SOLID, m_width, RGB(0, 0, 0));
    break;
}
dc.SelectObject(newPen);
dc.MoveTo(rect.left+10, rect.top+40);
dc.LineTo(rect.right-10, rect.top+40);
```

绘图程序的例子(10)

■ 在CTestView类定义中

```
private:  
    int m_width;  
    int m_style;  
    COLORREF m_color;
```

■ 在CTestView构造函数中

```
m_width=1;  
m_style=0;  
m_color=RGB(0, 0, 0);
```

绘图程序的例子(11)

- 在CTestView::OnViewPen() 中

```
CPenDialog penDlg;  
penDlg.m_width=m_width;  
penDlg.m_style=m_style;  
if (penDlg.DoModal()==IDOK)  
{ m_width=penDlg.m_width;  
  m_style=penDlg.m_style; }
```

- 在CTestView::OnViewColor() 中

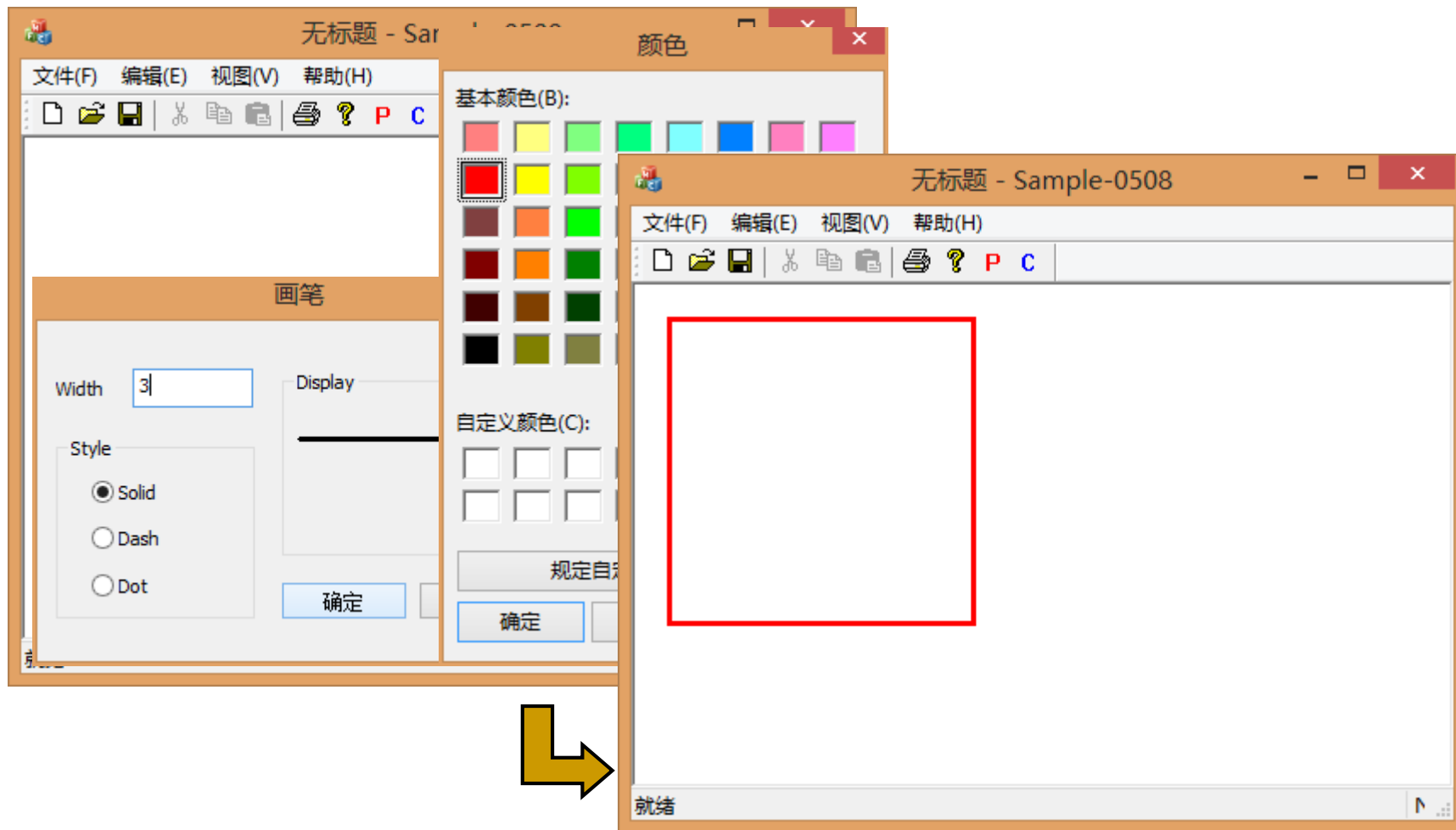
```
CColorDialog colorDlg;  
if (colorDlg.DoModal()==IDOK)  
    m_color=colorDlg.GetColor();
```

绘图程序的例子(12)

■ 在CTestView::OnLButtonDown() 中

```
CPen newPen;  
switch(m_style) {  
case 0:  
    newPen.CreatePen(PS_SOLID, m_width, m_color); break;  
case 1:  
    newPen.CreatePen(PS_DASH, m_width, m_color); break;  
case 2:  
    newPen.CreatePen(PS_DOT, m_width, m_color); break; }  
CDC* pDC=GetDC();  
pDC->SelectObject(newPen);  
pDC->Rectangle(20, 20, 200, 200);
```


绘图程序的例子(13)



旋转框控件(1)

- 旋转框(Spin)控件与编辑框控件一起使用，通常用于双向计数器
 - ✓ 添加编辑和旋转框，编辑框的Tab次序在旋转框前一个
 - ✓ Auto Buddy和Set Buddy Integer设为true
 - ✓ Alignment设为Right
 - ✓ Wrap和Arrow Keys设为true

旋转框控件 (2)

- CSpinButtonCtrl类用于管理Spin控件
 - ✓ SetRange() 用于设置取值范围
 - ✓ SetPos() 用于设置当前值
- 在CTestDialog::OnInitDialog() 中

```
m_ctrlSpin.SetRange(0, 50);  
m_ctrlSpin.SetPos(25);
```

滑动条控件(1)

- 滑动条(Slider)控件又称为游标控件，通常用于滑动游标取值
 - ✓ Auto Ticks和Tick Marks设为true
 - ✓ Orientation设为Horizontal
 - ✓ Point设为Both

滑动条控件(2)

- CSliderCtrl类用于管理Slider控件
 - ✓ SetRange() 用于设置取值范围
 - ✓ SetTicFreq() 用于设置取值幅度
 - ✓ SetPos() 用于设置当前值
 - ✓ SetLineSize() 用于设置方向键滑动值
 - ✓ SetPageSize() 用于设置Page键滑动值

滑动条控件 (3)

- 在CTestDialog::OnInitDialog() 中

```
m_ctrlSlider.SetRange(0, 10);  
m_ctrlSlider.SetTicFreq(1);  
m_ctrlSlider.SetPos(3);
```

- 在CTestDialog::OnHScroll() 中

```
if(m_ctrlSlider.GetPos()==10)  
    MessageBox(L"Slider to 10!");
```

进展条控件(1)



例5-9

- 进展条(Progress)控件又称为进度条，通常用于显示程序执行进展
 - ✓ Border设为true
 - ✓ Orientation设为Horizontal
 - ✓ Smooth设为true

进展条控件 (2)

- CProgressCtrl类用于管理Progress控件
 - ✓ SetRange(): 设置取值范围
 - ✓ SetStep(): 设置步长
 - ✓ SetPos(): 设置当前值
 - ✓ StepIt(): 按步长改变当前值
 - ✓ OffsetPos(): 按偏移量改变当前值

进展条控件 (3)

- 在CTestDialog::OnInitDialog() 中

```
m_ctrlProgress.SetRange(0, 100);  
m_ctrlProgress.SetStep(10);
```

- 在CTestDialog::OnTest() 中

```
for(int i=0;i<10;i++)  
{ m_ctrlProgress.StepIt();  
  Sleep(100); }  
if(m_ctrlProgress.GetPos()==100)  
  MessageBox(L"Progress to 100!");
```

进展条控件(4)

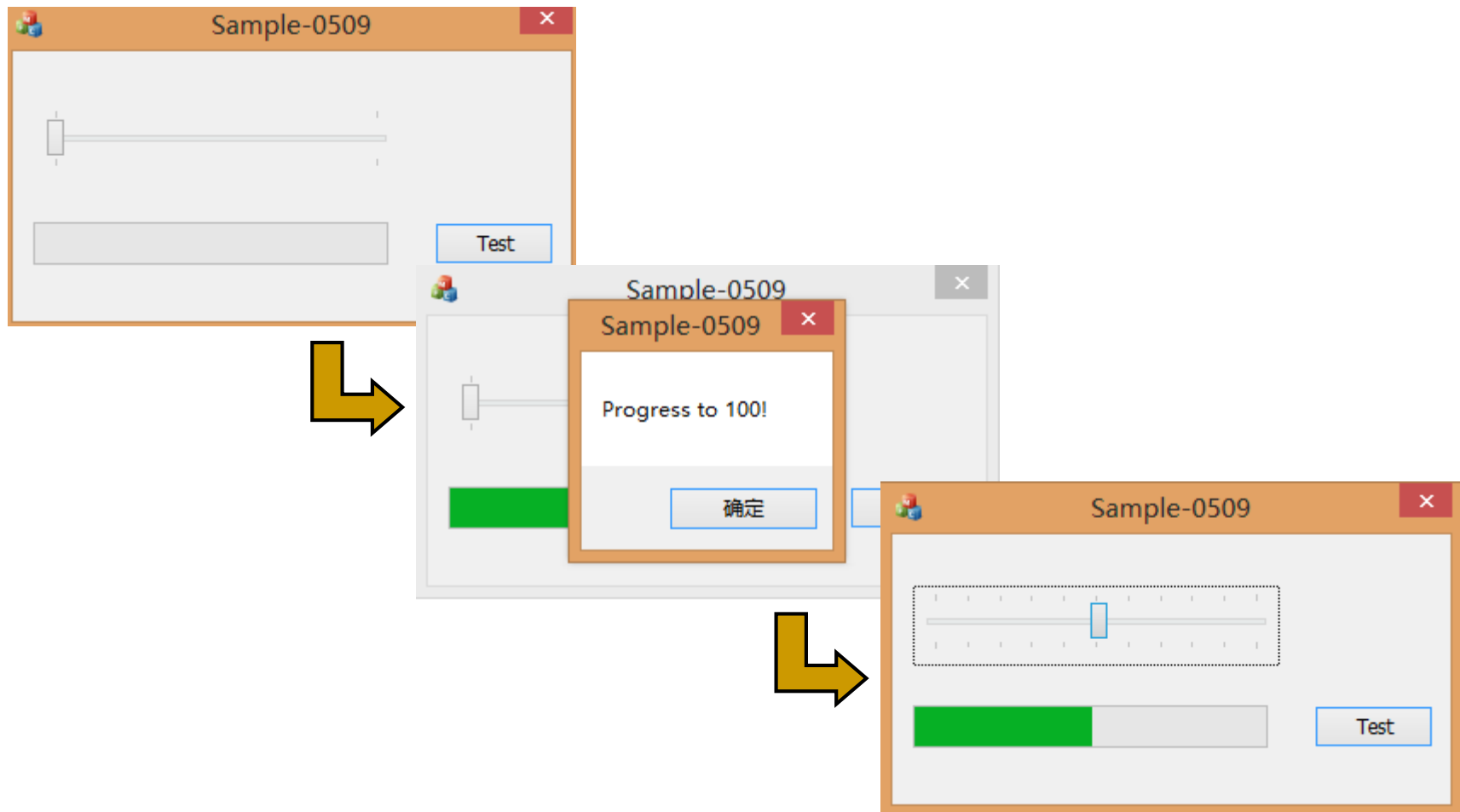
- 当滑动条或进展条滑动时，发送WM_VSCROLL或WM_HSCROLL消息给父窗口
- 在CTestDialog::OnInitDialog() 中

```
m_ctrlSlider.SetRange(0, 100);  
m_ctrlSlider.SetTicFreq(10);
```

- 在CTestDialog::OnHScroll() 中

```
m_ctrlProgress.SetPos(m_ctrlSlider.  
GetPos());
```

进展条控件 (5)



直接创建控件(1)

例5-10

- 在CTestView类定义中

```
private: CSliderCtrl m_ctrlSlider;
```

- 在CTestView类定义前

```
#define IDC_SLIDER 101
```

- 在CTestView::OnCreate() 中

```
m_ctrlSlider.Create(WS_CHILD|WS_VISIBLE|WS_BORDER|TBS_AUTOTICKS|TBS_BOTH|TBS_HORZ,  
CRect(40, 40, 300, 80), this, IDC_SLIDER);
```

直接创建控件 (2)

- 在CTestView::OnCreate() 中

```
m_ctrlSlider.SetRange(0, 100);  
m_ctrlSlider.SetTicFreq(10);
```

- 在CTestView::OnHScroll() 中

```
CSliderCtrl* pSlider=(CSliderCtrl*)  
pScrollBar;  
m_str.Format(L"%d", pSlider->GetPos());  
Invalidate(true);
```

- 在CTestView::OnDraw() 中

```
pDC->TextOutW(40, 20, m_str);
```

直接创建控件 (3)

- 在CTestView类定义中

```
private: CProgressCtrl m_ctrlProgress;
```

- 在CTestView类定义前

```
#define IDC_PROGRESS 102
```

- 在CTestView::OnCreate() 中

```
m_ctrlProgress.Create(WS_CHILD|WS_VISIBLE  
|WS_BORDER, CRect(400, 40, 700, 80), this, IDC_  
PROGRESS);
```

直接创建控件(4)

- 在CTestView::OnCreate() 中

```
m_ctrlProgress.SetRange(0, 100);  
m_ctrlProgress.SetStep(10);  
m_ctrlProgress.SetPos(50);
```

- 在CTestView::OnLButtonDown() 中

```
for(int i=0;i<5;i++)  
{ m_ctrlProgress.StepIt();  
  Sleep(100); }
```

哪个控件不受WM_HSCROLL或WM_VSCROLL消息的控制?

- ☐ A 滑动条控件
- ☐ B 进展条控件
- ☒ C 旋转框控件
- ☐ D 滚动条控件

动画控件(1)

- 动画(Animate)控件用于播放视频, 仅限于无声的avi文件
 - ✓ 属性有Auto Play、Center与Border
- CAnimateCtrl类用于管理Animate控件
 - ✓ 添加成员m_ctrlAnimate

动画控件 (2)

例5-11

■ 设计对话框控件

控件	控件标识	说明
Animate Box	IDC_ANIMATE	
Button	IDC_OPEN	Open
Button	IDC_PLAY	Play
Button	IDC_STOP	Stop
Button	IDC_DISPLAY	Display

动画控件 (3)

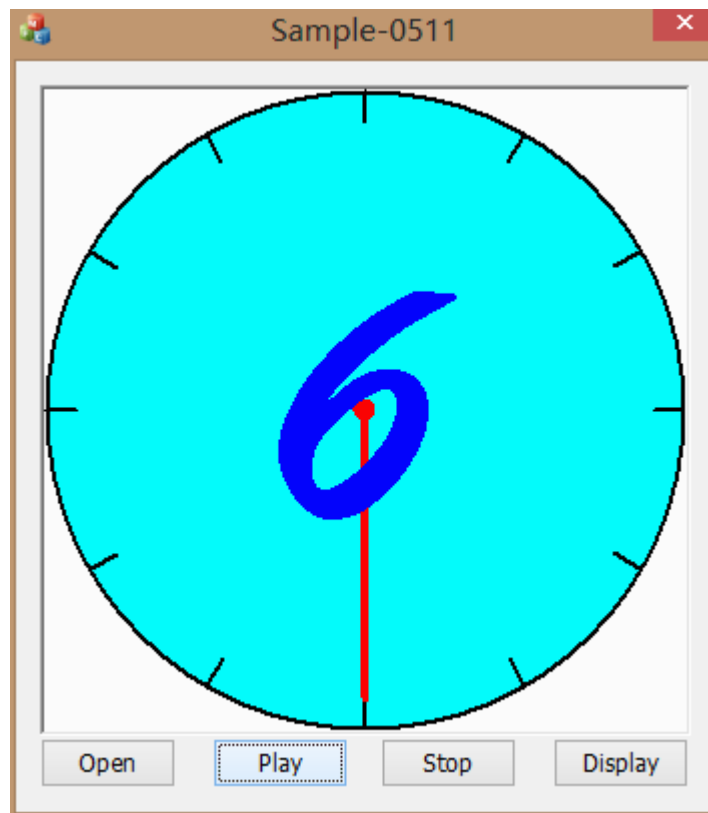
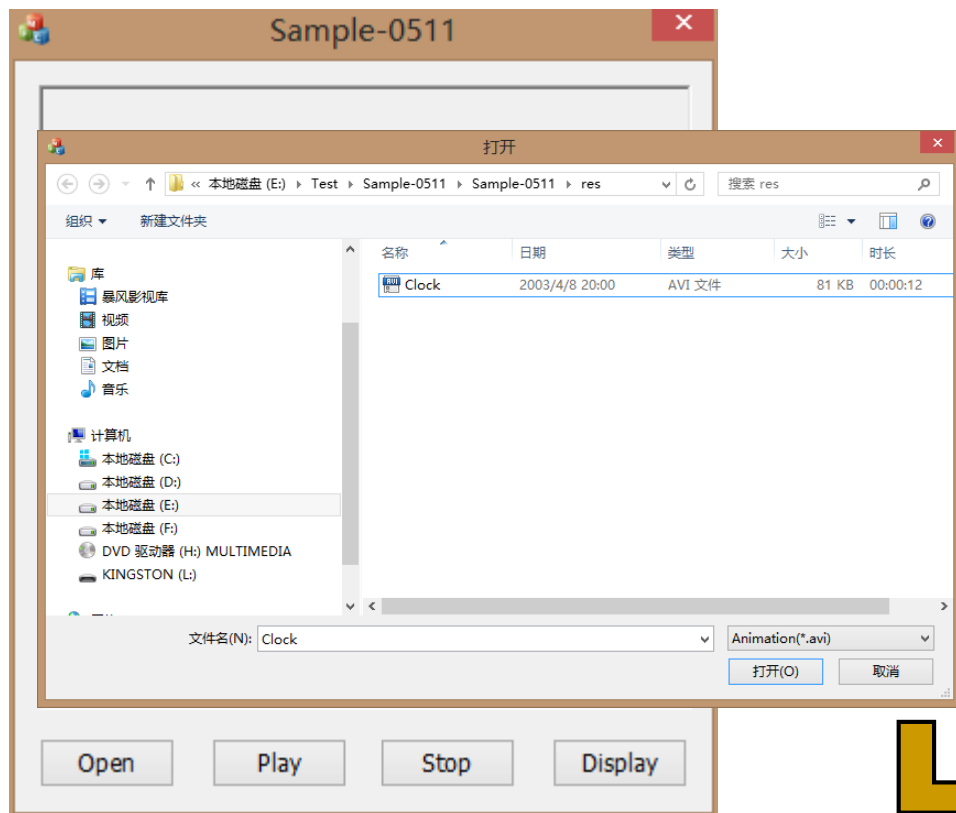
- 在CFileDialog::OnOpen() 中

```
CFileDialog dlg(true, NULL, NULL,  
OFN_READONLY, L"Animation (*.avi) | *.avi |");  
if (dlg.DoModal() == IDOK)  
{ CString name=dlg.GetPathName();  
  m_ctrlAnimate.Stop();  
  m_ctrlAnimate.Close();  
  m_ctrlAnimate.Open(name); }
```

动画控件 (4)

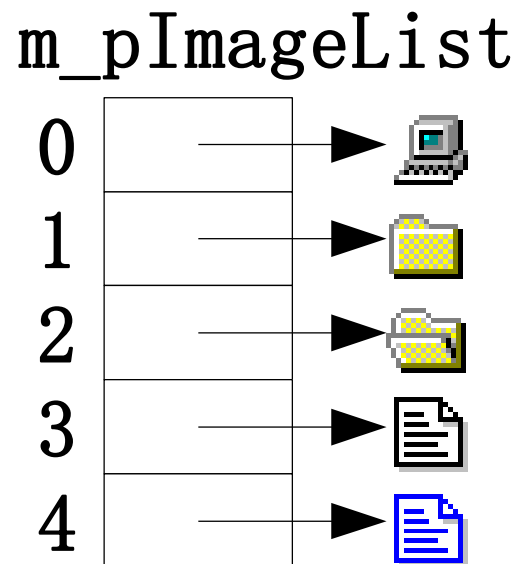
- 在CTestDialog::OnPlay() 中
`m_ctrlAnimate.Play(0, 11, 1);`
- 在CTestDialog::OnStop() 中
`m_ctrlAnimate.Stop();`
- 在CTestDialog::OnDisplay() 中
`m_ctrlAnimate.Seek(5);`

动画控件 (5)



图像列表与位图(1)

- 图像列表类似于位图数组，每个位图都有一个索引，用于查找特定图像
- `CImageList`类用于创建、显示和管理图像列表



图像列表与位图 (2)

例5-12

- 在Resource View中加载位图

IDB_CLASS、IDB_STUREC、IDB_COMPUTERDEPT、
IDB_SELECTEDCLASS与IDB_SELECTEDSTUREC

- 在CTestDialog类定义中

```
private:  
    CImageList* m_pImageList;  
    void AddBitmapToImageList(int num);
```

图像列表与位图(3)

■ 在CTestDialog构造函数中

```
m_pImageList=new CImageList();  
m_pImageList->Create(16, 16, true, 0, 0);  
AddBitmapToImageList(IDB_COMPUTERDEPT);  
AddBitmapToImageList(IDB_CLASS);  
AddBitmapToImageList(IDB_SELECTEDCLASS);  
AddBitmapToImageList(IDB_STUREC);  
AddBitmapToImageList(IDB_SELECTEDSTUREC);
```


图像列表与位图(4)

- 在CTestDialog::AddBitmapToImageList() 中

```
CBitmap bmp;  
bmp.LoadBitmap(num);  
m_pImageList->Add(&bmp, RGB(255, 255, 255));  
bmp.DeleteObject();
```

- 在CTestDialog::OnDestroy() 中

```
delete m_pImageList;
```

图像列表与位图 (5)

- 在CTestDialog::OnPaint()的else中

```
CPaintDC dc(this);  
CPoint pt1(10, 10), pt2(10, 60);  
int count=m_pImageList->GetImageCount();  
for(int i=0;i<count;i++)  
{ m_pImageList->Draw(&dc, i, pt1,  
ILD_NORMAL);  
    m_pImageList->Draw(&dc, i, pt2, ILD_MASK);  
    pt1.x+=50; pt2.x+=50; }
```

树状控件(1)

- 树状控件(Tree)通过树状结构显示信息
- 树的顶部称为父条目，其下条目称为子条目
- 用户可展开或折叠树中条目，以控制需要显示的信息
- 当需要更多显示条目时，树状控件会自动显示滚动条

树状控件 (2)

■ 树状控件的风格属性

- ✓ TVS_HASLINES: 在父/子结点间连线
- ✓ TVS_LINESATROOT: 在根/子结点间连线
- ✓ TVS_HASBUTTONS: 在结点前添加按钮, 表示结点是否被展开
- ✓ TVS_EDITLABELS: 结点内容可否编辑
- ✓ TVS_SHOWSELALWAYS: 失焦时显示当前结点
- ✓ TVS_DISABLEDRAHDROP: 不允许拖放

树状控件 (3)

■ 树状控件的成员函数

- ✓ SetImageList() 与 InsertItem()
- ✓ GetSelectedItem() 与 SelectItem()
- ✓ GetItemText() 与 SetItemText()
- ✓ DeleteItem() 与 DeleteAllItems()
- ✓ GetRootItem()、GetChildItem()、
GetParentItem() 与 GetNextSiblingItem()

树状控件(4)

■ 树状控件的消息映射

- ✓ TVN_SELCHANGED: 选中结点发生改变
- ✓ TVN_ITEMEXPANDED: 选中结点被展开
- ✓ TVN_BEGINLABELEDIT: 开始编辑结点内容
- ✓ TVN_ENDLABELEDIT: 结束编辑结点内容
- ✓ TVN_GETDISPINFO: 获得结点信息

树状控件 (5)

例5-13

■ 设计对话框控件

控件	控件标识	说明
Tree Control	IDC_TREE	m_ctrlTree
Static Text	IDC_STATIC	Selected Node
Edit Box	IDC_INFO	m_info

■ Tree Control风格

- ✓ Has Button(增加按钮)、Has Lines(增加父子连线)、Lines at root(增加根连线)

树状控件 (6)

- 在Resource View中加载位图文件

IDB_CLASS、IDB_STUREC、IDB_COMPUTERDEPT、
IDB_SELECTEDCLASS与IDB_SELECTEDSTUREC

- 在CTestDialog类定义中

```
private:  
    CImageList* m_pImageList;  
    void AddBitmapToImageList(int num);
```


树状控件(7)

■ 在CTestDialog类定义中

```
struct CStudent
{ CString id;  CString name;  int age;
  CStudent(CString sId, CString sName, int
nAge)
  { id=sId;  name=sName;  age=nAge; };
};
```

■ 在CTestDialog::AddBitmapToImageList() 中

```
CBitmap bmp;  bmp.LoadBitmap(num);
m_pImageList->Add(&bmp, RGB(255, 255, 255));
bmp.DeleteObject();
```

树状控件 (8)

- 在CTestDialog类构造函数中

```
m_pImageList=new CImageList();  
m_pImageList->Create(16, 16, true, 0, 0);  
AddBitmapToImageList(IDB_COMPUTERDEPT);  
AddBitmapToImageList(IDB_CLASS);  
AddBitmapToImageList(IDB_SELECTEDCLASS);  
AddBitmapToImageList(IDB_STUREC);  
AddBitmapToImageList(IDB_SELECTEDSTUREC);
```

树状控件 (9)

■ 在CTestDialog::OnInitDialog() 中

```
m_ctrlTree.SetImageList(m_pImageList, TVSIL_NORMAL);  
HTREEITEM root=m_ctrlTree.InsertItem(L"Computer  
Deptment", 0, 0);  
HTREEITEM layer=m_ctrlTree.InsertItem(L"Software",  
1, 2, root);  
HTREEITEM curNode;  
CStudent *pStudent;  
pStudent=new CStudent(L"1001", L"zhang", 18);  
curNode=m_ctrlTree.InsertItem(pStudent->name, 3, 4,  
layer);  
m_ctrlTree.SetItemData(curNode, (DWORD)pStudent);  
pStudent=new CStudent(L"1002", L"wang", 19);
```

树状控件(10)

```
curNode=m_ctrlTree.InsertItem(pStudent->name, 3, 4,
layer);
m_ctrlTree.SetItemData(curNode, (DWORD)pStudent);
layer=m_ctrlTree.InsertItem(L"Application", 1, 2,
root);
pStudent=new CStudent(L"1003", L"li", 20);
curNode=m_ctrlTree.InsertItem(pStudent->name, 3, 4,
layer);
m_ctrlTree.SetItemData(curNode, (DWORD)pStudent);
pStudent=new CStudent(L"1004", L"zhao", 21);
curNode=m_ctrlTree.InsertItem(pStudent->name, 3, 4,
layer);
m_ctrlTree.SetItemData(curNode, (DWORD)pStudent);
m_ctrlTree.Expand(root, TVE_EXPAND);
```

树状控件(11)

- 在CTestDialog::OnSelchangedTree() 中

```
HTREEITEM selNode=m_ctrlTree.GetSelectedItem();  
if(m_ctrlTree.GetItemData(selNode))  
{ CStudent* pStudent=(CStudent*)m_ctrlTree.  
  GetItemData(selNode);  
  m_info.Format(L"Id: %s\r\nName: %s\r\nAge: %d",  
pStudent->id, pStudent->name, pStudent->age);  
  UpdateData(false); }  
else  
{ m_info=m_ctrlTree.GetItemText(selNode);  
  UpdateData(false); }
```

树状控件(12)

■ 在CTestDialog::OnDestroy() 中

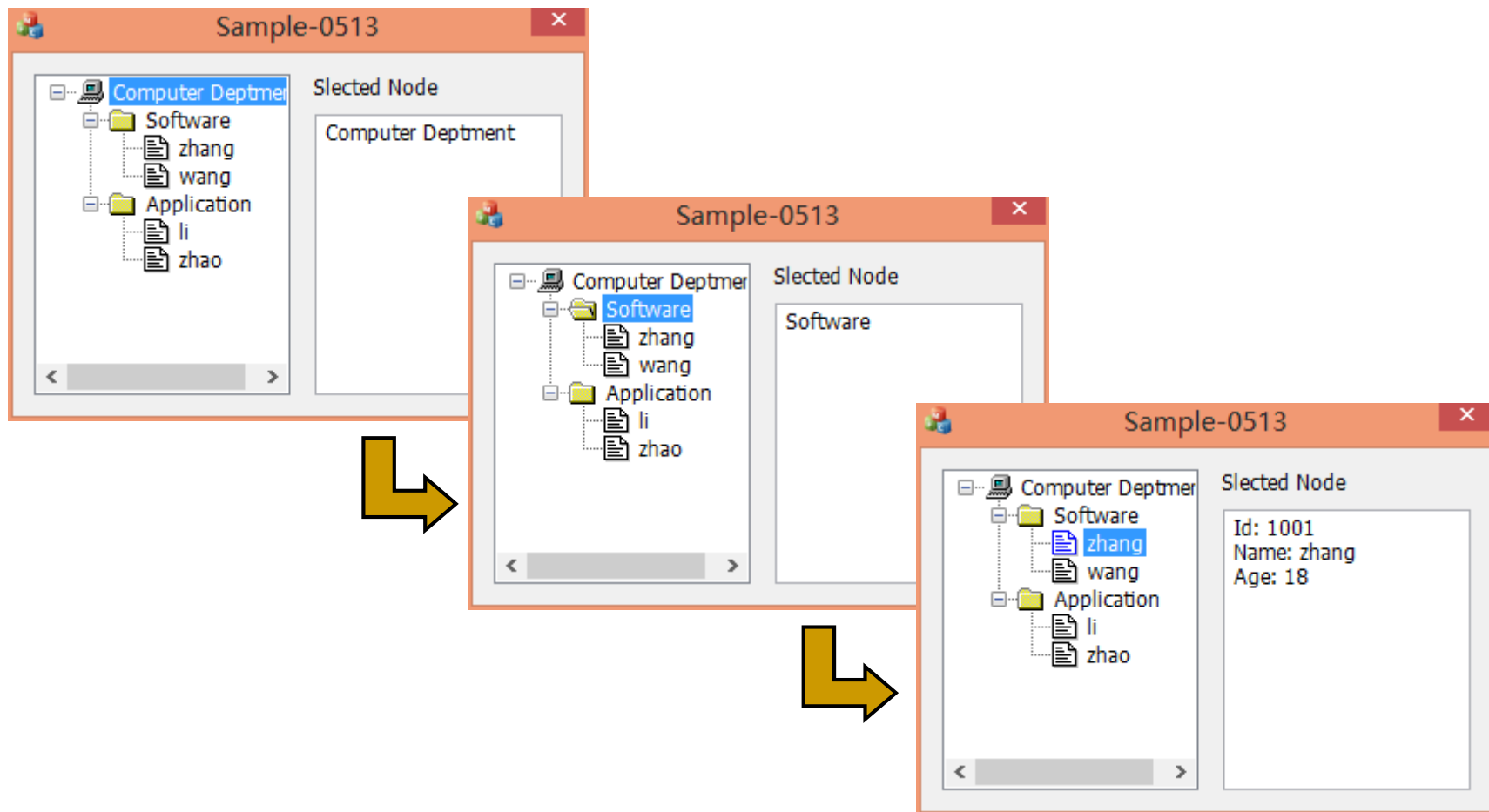
```
HTREEITEM rootNode=m_ctrlTree.GetRootItem();
HTREEITEM curLayerNode=m_ctrlTree.GetChildItem
(rootNode);
while(curLayerNode)
{
    HTREEITEM curStuNode=m_ctrlTree.GetChildItem
(curLayerNode);
    while(curStuNode)
    {
        CStudent* pStudent=(CStudent*)m_ctrlTree.
GetItemData(curStuNode);
```

树状控件(13)

■ 在CTestDialog::OnDestroy() 中

```
delete pStudent;
curStuNode=m_ctrlTree.GetNextSiblingItem
(curStuNode);
}
curLayerNode=m_ctrlTree.GetNextSiblingItem
(curLayerNode);
}
m_ctrlTree.DeleteAllItems();
delete m_pImageList;
```

树状控件(14)



树状控件拖放功能(1)

- 树状控件中实现拖放功能
 - ✓ TVN_BEGINDRAG: 通知树控件的父窗口拖动已开始
 - ✓ WM_MOUSEMOVE: 移动鼠标时发送
 - ✓ WM_LBUTTONDOWN: 释放鼠标左按钮时发送

树状控件拖放功能(2)

例5-14

■ 设计对话框控件

控件	控件标识	说明
Tree Control	IDC_TREE	m_ctrlTree

■ 设置树控件的风格

- ✓ Has Button (增加+、-按钮)
- ✓ Has Lines (增加父子连线)
- ✓ Lines at root (增加根连线)

树状控件拖放功能(3)

■ 在CTestDialog类定义中

```
private:  
    CImageList* m_pImageList;  
    BOOL m_drag;  
    HTREEITEM m_dragItem;  
    HTREEITEM m_dragTarget;  
    void AddBitmapToImageList(int num);  
    void InitTree();
```

树状控件拖放功能(4)

■ 在CTestDialog::InitTree()中

```
m_ctrlTree.SetImageList(m_pImageList, TVSIL_NORMAL);  
HTREEITEM root=m_ctrlTree.InsertItem(L"Computer  
Deptment", 0, 0);  
HTREEITEM layer=m_ctrlTree.InsertItem(L"Software",  
1, 2, root);  
m_ctrlTree.InsertItem(L"zhang", 3, 4, layer);  
m_ctrlTree.InsertItem(L"wang", 3, 4, layer);  
layer=m_ctrlTree.InsertItem(L"Application", 1, 2,  
root);  
m_ctrlTree.InsertItem(L"li", 3, 4, layer);  
m_ctrlTree.InsertItem(L"zhao", 3, 4, layer);  
m_ctrlTree.Expand(root, TVE_EXPAND);
```

树状控件拖放功能(5)

- 在CTestDialog::OnInitDialog() 中

```
m_drag=FALSE;  
m_dragTarget=NULL;  
m_dragItem=NULL;  
InitTree();
```

- 在CTestDialog::OnDestroy() 中

```
m_ctrlTree.DeleteAllItems();  
delete m_pImageList;
```

树状控件拖放功能(6)

- 在CTestDialog::OnBeginDragTree() 中

```
m_dragItem=pNMTreeView->itemNew.hItem;
int layer=0;  //被拖动结点层数
HTREEITEM item=m_dragItem;
while(item)
{ item=m_ctrlTree.GetParentItem(item);
  layer++; }
if(layer==3)  //学生结点
{ CImageList* pDragImage;
  pDragImage=m_ctrlTree.CreateDragImage
(m_dragItem);
```

树状控件拖放功能(8)

```
m_ctrlTree.SelectItem(m_dragItem);  
pDragImage->BeginDrag(0, CPoint(0, 0));  
pDragImage->DragEnter(&m_ctrlTree,  
pNMTreeView->ptDrag);  
SetCapture();  
m_drag=true;  
delete pDragImage;  
}
```

树状控件拖放功能(9)

- 在CTestDialog::OnMouseMove() 中

```
if(m_drag)
{ CPoint pt(point);
  MapWindowPoints(&m_ctrlTree, &pt, 1);
  CImageList::DragMove(pt);
  UINT hitTest=TVHT_ONITEM;
  m_dragTarget=m_ctrlTree.HitTest(pt,
  &hitTest); }
```


树状控件拖放功能(10)

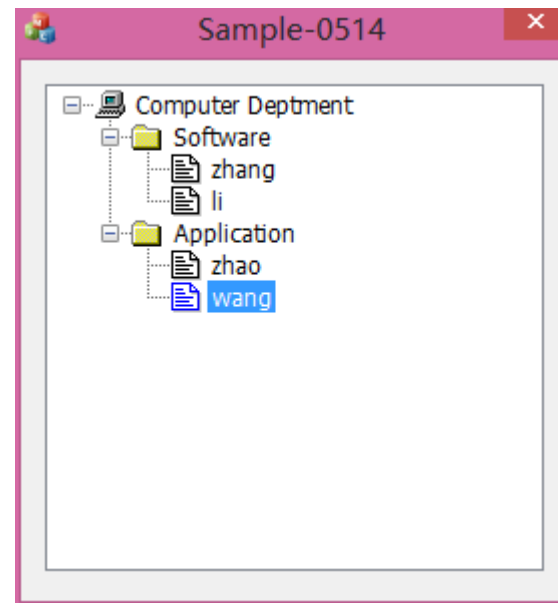
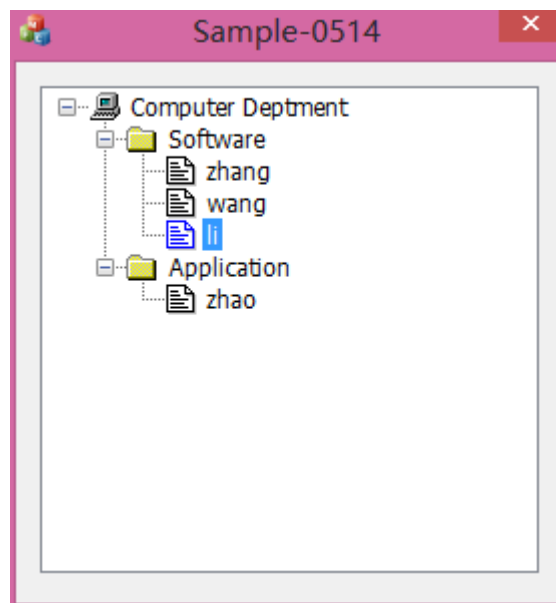
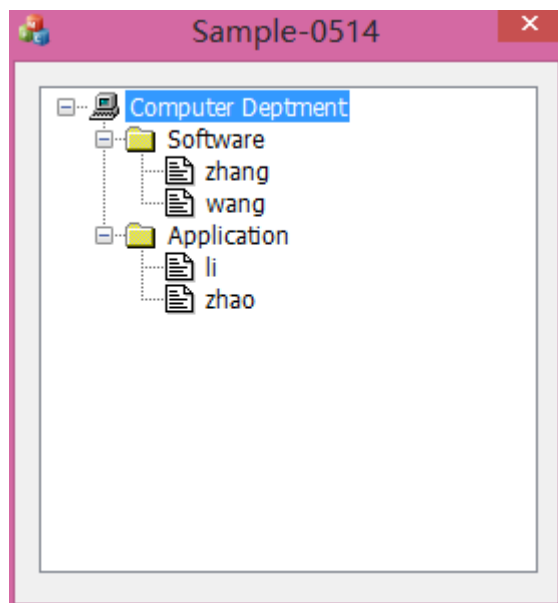
- 在CTestDialog::OnLButtonUp() 中

```
if(m_drag)
{ CImageList::DragLeave(&m_ctrlTree);
  CImageList::EndDrag();
  ReleaseCapture();
  m_drag=FALSE;  //获得目标位置层数
  int layer=0;
  HTREEITEM item=m_dragTarget;
  while(item)
  { item=m_ctrlTree.GetParentItem(item);
    layer++; }
  if(layer==3) { HTREEITEM parent;
```

树状控件拖放功能(11)

```
    parent=m_ctrlTree.GetParentItem  
(m_dragTarget);  
    CString label=m_ctrlTree.GetItemText  
(m_dragItem);  
    int image,selectedImage;  
    m_ctrlTree.GetItemImage(m_dragItem,  
image,selectedImage);  
    m_ctrlTree.InsertItem(label, image,  
selectedImage, parent, m_dragTarget);  
    m_ctrlTree.DeleteItem(m_dragItem); }  
}
```

树状控件拖放功能(12)



哪个属性决定树状控件的结点内容是否可编辑？

- ☐ A Has Lines
- ☒ B Edit Labels
- ☐ C Lines At Root
- ☐ D Disable Drag Drop

提交

第5次作业

- 设计一个单文档边框窗口程序，要求至少具有以下功能：
 - ✓ 程序具有打开对话框功能
 - ✓ 对话框至少包含8种控件，填写个人信息(姓名、性别、年龄、学号、籍贯、照片等)，初始化提供默认信息
 - ✓ 点击确定按钮，在视图中显示信息
 - ✓ 通过菜单项或工具栏按钮，保存与打开信息



谢谢大家

