# 这一节我们继续来学习列表框控件

# 学习大纲

|  |
| --- |
|  |

# 上一节剩下两个内容没有学习，我们这一节来学习一下

# 演练

## 1.把上一节的项目复制一份，改名Lesson32-listboxdemo

|  |
| --- |
|  |

## 2.我们把对话框拉大一点，然后添加一个列表框，在两个列表框之间添加一个按钮，点击按钮把坐标的列表项移动到右边的列表框中,ID为IDC\_BTN\_MOVE

|  |
| --- |
|  |

## 3.然后我们给这个按钮添加点击事件处理代码，注意需要 定义一下变量来辅助功能的实现，注意这里代码在上面的基础上做了一些微调，添加一些变量和删除了一些变量

|  |
| --- |
|  |

### 然后添加按钮点击代码

|  |
| --- |
|  |

### 效果：选中左边列表框里面的一项，点击中间的按钮就可用复制到右边

|  |
| --- |
|  |
|  |

### 注意:列表框也可以支持多选，需要把在资源视图里面把它的Selection改为Multiple

|  |
| --- |
|  |

### 效果，

|  |
| --- |
|  |

## 4.我们把左边的还是恢复到单选，把右边的设置为多选，为了方便操作我们在对话框的WM\_INITDIALOG消息处理代码这添加一些列表项，也是需要一些辅助变量的。

|  |
| --- |
|  |
|  |
|  |
|  |

### 初始化后的效果

|  |
| --- |
|  |

## 5.然后我们添加一个按钮，点击按钮选中列表框2里面的多个项，因为它已经设置为可用多选

|  |
| --- |
|  |

## 6.我们给这个按钮添加点击事件处理代码，其实就是发送LB\_SETSEL消息，需要一个一个来选择，可用发送一次，选择一个项，也可以发送多次，选择多个项，可用连续也可以不连续

|  |
| --- |
|  |

## 7.我们来学习一下全部选中功能，给对话框添加一个按钮文本为：全部选中，ID为：IDC\_BTN\_ALL\_SEL

|  |
| --- |
|  |

## 8.我们来给这个按钮添加点击事件处理代码，其实就是发送一个LB\_SETSEL消息，把lParam设置为-1，就可用了

|  |
| --- |
|  |

## 9.我们给对话框添加一个取消全部按钮，就是取消选中全部，ID为ID\_BTN\_ALL\_UNSEL

|  |
| --- |
|  |

## 10.然后我们给他添加点击事件处理代码，其实就是上面的代码把wParam设置为0，表示不选中，把lParam设置为-1表示全部，也就是全不选

|  |
| --- |
|  |

### 效果：先点击选中全部

|  |
| --- |
|  |

### 再点击取消全部

|  |
| --- |
|  |

### 其实，不管你选择多少项，点击取消全部都会全部不选中，如我们选中3项

|  |
| --- |
|  |

### 然后点击取消全部，效果如下

|  |
| --- |
|  |

## 11.下面我们来连续移动多个选中项目到另外一个列表框，我们添加一个列表框和一个按钮，调整布局如下,把新按钮的id改为: IDC\_BTN\_MOVE\_MULTI

|  |
| --- |
|  |
|  |

## 12.然后我们来给这个按钮添加点击事件处理代码，需要使用循环，首先我们需要知道选中了多少个，需要一个变量来帮助，叫做selCount，还需要一个ListBox3对应的窗口句柄

|  |
| --- |
|  |
| 需要获取这个ListBox3的句柄 |
|  |

## 13.然后我们来完成代码，我们还需要一个变量isSelected

|  |
| --- |
|  |

## 14.我们继续写这个按钮的代码

|  |
| --- |
|  |

# 这一节我们的学习到此为止，完整的代码如下

## listboxdemo.cpp

|  |
| --- |
| #include<Windows.h>  #include"resource.h"  INT\_PTR CALLBACK DlgProc(HWND hDlg,UINT uMsg,WPARAM wParam,LPARAM lParam)  {  BOOL ret = TRUE;  int wmId,wmEvent;  static HWND hwndLB,hwndEdit,hwndDel,hwndLB2,hwndLB3;  TCHAR szItem[128];  TCHAR szItem2[128];  TCHAR \*pthContent;  static int curSel,isSelected;  static int count,length,selCount;  wmId = LOWORD(wParam);  wmEvent = HIWORD(wParam);  hwndLB = GetDlgItem(hDlg,IDC\_LIST1);  hwndEdit = GetDlgItem(hDlg,IDC\_EDIT1);  hwndDel = GetDlgItem(hDlg,IDC\_BTN\_DEL);  hwndLB2 = GetDlgItem(hDlg,IDC\_LIST2);  hwndLB3 = GetDlgItem(hDlg,IDC\_LIST3);    switch(uMsg)  {  case WM\_INITDIALOG:  lstrcpy(szItem2,"好好学习");  SendDlgItemMessage(hDlg,IDC\_LIST2,LB\_ADDSTRING,0,(LPARAM)szItem2);  ZeroMemory(szItem2,128);  lstrcpy(szItem2,"天天向上");  SendDlgItemMessage(hDlg,IDC\_LIST2,LB\_ADDSTRING,0,(LPARAM)szItem2);  ZeroMemory(szItem2,128);  lstrcpy(szItem2,"恭喜发财");  SendDlgItemMessage(hDlg,IDC\_LIST2,LB\_ADDSTRING,0,(LPARAM)szItem2);  ZeroMemory(szItem2,128);  lstrcpy(szItem2,"红包拿来");  SendDlgItemMessage(hDlg,IDC\_LIST2,LB\_ADDSTRING,0,(LPARAM)szItem2);  ZeroMemory(szItem2,128);  lstrcpy(szItem2,"谢谢你哟");  SendDlgItemMessage(hDlg,IDC\_LIST2,LB\_ADDSTRING,0,(LPARAM)szItem2);  break;  case WM\_CLOSE:  PostQuitMessage(0);  break;  case WM\_COMMAND:  /\*wmId = LOWORD(wParam);  wmEvent = HIWORD(wParam);  hwndLB = GetDlgItem(hDlg,IDC\_LIST1);  hwndEdit = GetDlgItem(hDlg,IDC\_EDIT1);  hwndDel = GetDlgItem(hDlg,IDC\_BTN\_DEL);  hwndLB2 = GetDlgItem(hDlg,IDC\_LIST2);\*/  switch(wmId)  {  case IDOK:  if(IDOK == MessageBox(hDlg,TEXT("退出程序？"),TEXT("确认"),MB\_OKCANCEL))  {  EndDialog(hDlg,FALSE);  }  break;  case IDC\_BTN\_ADD:  GetDlgItemText(hDlg,IDC\_EDIT1,szItem,128);  if(lstrlen(szItem)==0)//当用户没有在编辑框里面输入直接点击按钮，弹出消息框，并且设置焦点  {    MessageBox(hDlg,TEXT("请输入需要添加的内容"),TEXT("输入内容"),MB\_OK);  SetFocus(hwndEdit);  return -1;  }    SendDlgItemMessage(hDlg,IDC\_LIST1,LB\_ADDSTRING,0,(LPARAM)szItem);//添加，有排序功能  //SendDlgItemMessage(hDlg,IDC\_LIST1,LB\_INSERTSTRING,0,(LPARAM)szItem);//插入  SetDlgItemText(hDlg,IDC\_EDIT1,TEXT(""));  SetFocus(hwndEdit);//个列表框添加了字符串后，编辑框需要重新设置输入焦点  break;  case IDC\_LIST1: //处理列表框消息  switch(wmEvent)  {  case LBN\_SELCHANGE://当用户选中一个列表框项，发送此消息  curSel = SendMessage(hwndLB,LB\_GETCURSEL,0,0);//获取当前选中的项  EnableWindow(hwndDel,TRUE);//选中了列表框项按钮才可用  break;  }  break;  case IDC\_BTN\_DEL:  SendMessage(hwndLB,LB\_DELETESTRING,curSel,0);  SetFocus(hwndEdit);  break;  case IDC\_BTN\_CLEAR:  SendMessage(hwndLB,LB\_RESETCONTENT,0,0);//清空列表框  SetFocus(hwndEdit);//清空后需要使得编辑框获得输入焦点  break;  case IDC\_BTN\_FIRST:  SendMessage(hwndLB,LB\_SETCURSEL,0,0);//清空列表框  SetFocus(hwndEdit);//选中后需要使得编辑框获得输入焦点  break;  case IDC\_BTN\_LAST:  count = SendMessage(hwndLB,LB\_GETCOUNT,0,0);  SendMessage(hwndLB,LB\_SETCURSEL,(count-1),0);//清空列表框  SetFocus(hwndEdit);//选中后需要使得编辑框获得输入焦点  break;  case IDC\_BTN\_CLEAR\_SEL:  SendMessage(hwndLB,LB\_SETCURSEL,-1,0);//清空列表框，需要把wParam设置为-1  SetFocus(hwndEdit);//取消选中后需要使得编辑框获得输入焦点  break;  case IDC\_BTN\_MOVE:  curSel = SendMessage(hwndLB,LB\_GETCURSEL,0,0);  length = SendMessage(hwndLB,LB\_GETTEXTLEN,curSel,0)+1;  //pthContent = new TCHAR[length];  pthContent =(TCHAR\*)calloc(length,sizeof(TCHAR));  SendMessage(hwndLB,LB\_GETTEXT,curSel,(LPARAM)pthContent);//获取左边内容  //MessageBox(hDlg,szContent,TEXT("内容"),MB\_OK);  //SendMessage(hwndLB2,LB\_ADDSTRING,0,(LPARAM)pthContent);  SendMessage(hwndLB2,LB\_INSERTSTRING,-1,(LPARAM)pthContent);//也可以这么写  //移动转换，需要把左边的对应项删除  SendMessage(hwndLB,LB\_DELETESTRING,curSel,0);  SetFocus(hwndEdit);//移动项目后需要使得编辑框获得输入焦点  break;  case IDC\_BTN\_MULTI\_SEL:  SendMessage(hwndLB2,LB\_SETSEL,1,0);//wParam为1表示选择，为0表示不选中  SendMessage(hwndLB2,LB\_SETSEL,1,2);//lParam是对应项是下标  SendMessage(hwndLB2,LB\_SETSEL,1,4);  break;  case IDC\_BTN\_ALL\_SEL:  SendMessage(hwndLB2,LB\_SETSEL,1,-1);//全选  //  break;  case IDC\_BTN\_ALL\_UNSEL:  SendMessage(hwndLB2,LB\_SETSEL,0,-1);//取消选中全部  break;  case IDC\_BTN\_MOVE\_MULTI:  //获取列表框项一共有多少个  selCount = SendMessage(hwndLB2,LB\_GETCOUNT,0,0);  //遍历所有项判断每一项是否被选中，注意这里最好使用倒序循环，  //使用顺序的话可能不能移动全部因为删除列表项后，索引会变，又会从0开始  for(int i=selCount-1;i>=0;i--)  {  isSelected = SendMessage(hwndLB2,LB\_GETSEL,i,0);  //如果被选中  if(isSelected !=0)  {  //获取对应项的字符串长度,需要+1用来存放'\0'  length = SendMessage(hwndLB2,LB\_GETTEXTLEN,i,0)+1;  pthContent =(TCHAR\*)calloc(length,sizeof(TCHAR)); //工具长度分配控件  ZeroMemory(pthContent,length);//初始化为0  SendMessage(hwndLB2,LB\_GETTEXT,i,(LPARAM)pthContent);//获取文本到缓冲区，  //让列表框3添加项,此时wParam不能传递-1，否则顺序会反过来  SendMessage(hwndLB3,LB\_INSERTSTRING,0,(LPARAM)pthContent);  SendMessage(hwndLB2,LB\_DELETESTRING,i,0);//然后删除列表框2对应的项  }  }  break;  default:  ret = FALSE;  break;  }  break;  default:  ret = FALSE;  break;  }  return ret;  }  int APIENTRY WinMain(HINSTANCE hInstance,HINSTANCE prev,LPTSTR lpcmd,int cmdshow)  {  DialogBox(hInstance,MAKEINTRESOURCE(IDD\_DIALOG1),NULL,DlgProc);  return 0;  } |

### 这一节的代码比较多，需要好好体会多多练习，列表框是一个比较重要的控件。需要好好掌握。

## 这里有一个需要注意的地方:如果了需要移动多个列表项,你需要先获取所有的项,然后逐个判断是否被选中,如果选中就移动.前提是这个列表框支持多选.需要进行设置

## 扩展:vs2019版本

|  |
| --- |
|  |
|  |
|  |

## listboxapp.cpp代码如下

|  |
| --- |
| #include<Windows.h>  #include"resource.h"  INT\_PTR CALLBACK DlgProc(HWND,UINT,WPARAM,LPARAM);  int APIENTRY WinMain(HINSTANCE hinstance, HINSTANCE prev, LPSTR lpCmd, int nshow)  {  DialogBox(hinstance, MAKEINTRESOURCE(IDD\_DIALOG1), NULL, DlgProc);  return 0;  }  INT\_PTR CALLBACK DlgProc(HWND hdlg, UINT msg, WPARAM wParam, LPARAM lParam)  {  BOOL ret = TRUE;  BOOL selected = FALSE;  static int index,count,len; //列表框控件选中项的索引  static HWND hwndEdit1, hwndList1, hwndList2, hwndList3;  hwndList1 = GetDlgItem(hdlg, IDC\_LIST1);  hwndList2 = GetDlgItem(hdlg, IDC\_LIST2);  hwndList3 = GetDlgItem(hdlg, IDC\_LIST3);  hwndEdit1 = GetDlgItem(hdlg, IDC\_EDIT1);  static WCHAR szContent[MAX\_PATH],\*szItem;  ZeroMemory(szContent, MAX\_PATH);  int cmdid = LOWORD(wParam);  switch (msg)  {  //case WM\_KEYDOWN: //这个消息的wParam是虚拟键  // /\*if (wParam == VK\_RETURN)  // {  // MessageBox(hdlg, L"Enter键被按下", L"消息", 0);  // }\*/  // MessageBox(hdlg, L"按键被按下", L"消息", 0);  // break; //win32创建的对话框默认是不响应 WM\_KEYDOWN的,写入还是白写.  case WM\_CLOSE:  EndDialog(hdlg, 0);  break;  case WM\_COMMAND:  switch (cmdid)  {  case IDC\_LIST1:  switch (HIWORD(wParam))  {  case LBN\_SELCHANGE: //选中了才允许删除  index = SendMessage(hwndList1, LB\_GETCURSEL, 0, 0);//获取选中项  EnableWindow(GetDlgItem(hdlg, IDC\_BTN\_DEL),TRUE);  EnableWindow(GetDlgItem(hdlg, IDC\_BTN\_MOVE), TRUE);//选中了,才允许移动  break;  }  break;  case IDC\_BTN\_ADD:  GetDlgItemText(hdlg, IDC\_EDIT1, szContent, MAX\_PATH);  if (lstrlen(szContent)==0)  {  MessageBox(hdlg, L"请输入要添加的内容", L"输入内容", 0);  SetFocus(hwndEdit1);  }  else  {  SendMessage(hwndList1,LB\_ADDSTRING,0,(LPARAM)szContent);  SetDlgItemText(hdlg, IDC\_EDIT1,L"");//清空内容防止重复添加  SetFocus(hwndEdit1);  }  break;  case IDC\_BTN\_CLEAR: //清空所有列表项  SendMessage(hwndList1, LB\_RESETCONTENT, 0, 0);  SetFocus(hwndEdit1);  EnableWindow(GetDlgItem(hdlg, IDC\_BTN\_DEL), FALSE);  break;  case IDC\_BTN\_DEL:  /\*index = SendMessage(hwndList1, LB\_GETCURSEL, 0, 0);\*/ //在列表框调整消息里面获取选中项比较合理  SendMessage(hwndList1, LB\_DELETESTRING, index, 0);  break;  case IDC\_BTN\_FIRST:  SendMessage(hwndList1, LB\_SETCURSEL, 0, 0); //wParam==0表示选中第一项  break;  case IDC\_BTN\_LAST:  count = SendMessage(hwndList1, LB\_GETCOUNT, 0, 0); //获取列表项的数量  SendMessage(hwndList1, LB\_SETCURSEL, (count - 1), 0); //wParam==0表示选中第一项  break;  case IDC\_BTN\_CLEAR\_SEL:  SendMessage(hwndList1, LB\_SETCURSEL, -1, 0);//取消选中需要发送 LB\_SETCURSEL,wParam为-1  break;  case IDC\_BTN\_MOVE:  if ((count = SendMessage(hwndList1, LB\_GETCOUNT, 0, 0)) != 0) //获取列表项的数量)  {  index = SendMessage(hwndList1, LB\_GETCURSEL, 0, 0);//获取当前选项  len = SendMessage(hwndList1, LB\_GETTEXTLEN, index, 0);//获取文本长度  szItem = (WCHAR\*)calloc(len+1, sizeof(WCHAR));  SendMessage(hwndList1, LB\_GETTEXT, index, (LPARAM)szItem);//获取文本  SendMessage(hwndList2, LB\_ADDSTRING, 0, (LPARAM)szItem);//把文本内容发送给list2  SendMessage(hwndList1, LB\_DELETESTRING, index, 0);//删除list1的当前项  SetFocus(hwndEdit1);//给编辑框设置输入焦点.  EnableWindow(GetDlgItem(hdlg, IDC\_BTN\_MOVE), FALSE);//防止用户在list1没有选中的时候点击按钮  free(szItem);  }  break;  case IDC\_BTN\_ALL\_SEL: //全部选择  SendMessage(hwndList2, LB\_SETSEL, 1, -1);//选择全部也是LB\_SETSEL消息此时wParam是1,lParam是-1表示全部  break;  case IDC\_BTN\_ALL\_UNSEL:  SendMessage(hwndList2, LB\_SETSEL, 0, -1);  break;  case IDC\_BTN\_MOVE\_MULTI://移动多个项目,需要使用循环,先获取所有的项,然后逐个判断是否选中,是就移动  //获取一共有多少项,  count = SendMessage(hwndList2, LB\_GETCOUNT, 0, 0);  //倒序循环  for (int i = count - 1; i >= 0 ; i--)  {  selected = SendMessage(hwndList2, LB\_GETSEL, i, 0);  if (selected)  {  len = SendMessage(hwndList2, LB\_GETTEXTLEN, i, 0);  szItem = (WCHAR\*)calloc(len, sizeof(WCHAR));  SendMessage(hwndList2, LB\_GETTEXT, i, (LPARAM)szItem);//获取list2的文本  SendMessage(hwndList3, LB\_ADDSTRING, 0, (LPARAM)szItem);//把它添加到list3  SendMessage(hwndList2, LB\_DELETESTRING, i, 0);//然后删除list2选中项的文本  // free(szItem);  }  }  break;  default:  ret = FALSE;  break;  }  break;  default:  ret = FALSE;  break;  }  return ret;  } |

### 注意:win32对话框程序是不能够响应WM\_KEYDOWN消息的