# 这一节我们来学习列表框控件

# 学习大纲

|  |
| --- |
|  |

## SetFocus函数的使用

|  |
| --- |
| **函数原型：HWND SetFocus（HWND hWnd）；      参数：      hWnd：接收键盘输入的窗口指针。若该参数为NULL，则击键被忽略。      返回值：若函数调用成功，则返回原先拥有键盘焦点的窗口句柄。若hWnd参数无效或窗口未与调用线程的消息队列相关，则返回值为NULL。若要获得更多错误信息，可以调用GetLastError函数。      备注：SetFocus函数发送WM\_KILLFOCUS消息到失去键盘焦点的窗口，并且发送WM\_SETFOCUS消息到接受键盘焦点的窗口。它也激活接受键盘焦点的窗口或接受键盘焦点的窗口的父窗口。      若一个窗口是活动的，但没有键盘焦点，则任何按键将会产生WM\_SYSCHAR,WM\_SYSKEYDOWN或WM\_SYSKEYUP消息。若VK\_MENU键也被按下，则消息的IParam参数将设置第30位。否则，所产生的消息将不设置此位。      使用AttachThreadInput函数，一个线程可将输入处理连接到其他线程。这使得线程可以调用SetFocus函数为一个与其他线程的消息队列相关的窗口设置键盘焦点。** |

# 老师的列表框项选中的处理代码

|  |
| --- |
|  |

# 演练

## 1.新建一个c++常规空项目，取名：Lesson31-listboxdemo，然后新建一个cpp文件叫做listboxdemo

|  |
| --- |
|  |

## 2.在注意文件的文件夹上面点击右键-》添加-》资源，然后选中Dialog，点击右边的新建按钮，就会创建一个对话框资源

|  |
| --- |
|  |

## 3，先把取消按钮删除，然后把确定按钮的文本改为退出，然后在cpp文件里面导入开发工具创建的resource.h文件，定义一个对话框过程函数，在里面添加对用户按下回车键的处理代码和用户点击x按钮的消息学会了代码。然后我们自己编写WinMain函数在里面利用这个对话框资源创建一个对话框,下面就是基本代码

|  |
| --- |
| #include<Windows.h>  #include"resource.h"  INT\_PTR CALLBACK DlgProc(HWND hDlg,UINT uMsg,WPARAM wParam,LPARAM lParam)  {  BOOL ret = TRUE;  switch(uMsg)  {  case WM\_CLOSE:  PostQuitMessage(0);  break;  case WM\_COMMAND:  switch(LOWORD(wParam))  {  case IDOK:  if(IDOK == MessageBox(hDlg,TEXT("退出程序？"),TEXT("确认"),MB\_OKCANCEL))  {  EndDialog(hDlg,FALSE);  }  break;  default:  ret = FALSE;  }  break;  default:  ret = FALSE;  }  return ret;  }  int APIENTRY WinMain(HINSTANCE hInstance,HINSTANCE prev,LPTSTR lpcmd,int cmdshow)  {  DialogBox(hInstance,MAKEINTRESOURCE(IDD\_DIALOG1),NULL,DlgProc);  return 0;  } |

## 4.然后我们开始添加控件，先添加一个列表框ListBox控件，使用默认ID，然后我们在滤波控件上面添加一个静态文本一个编辑框和一个Add按钮，按的ID为IDC\_BTN\_ADD我们给按钮添加点击事件处理代码使得点击按钮就把编辑框的内容添加到列表框,注意：如果想程序一启动，编辑框就有输入焦点，需要把编辑框的tab顺序改为1.

|  |
| --- |
|  |
| #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;  TCHAR szItem[128];  switch(uMsg)  {  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);  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;  default:  ret = FALSE;  }  break;  default:  ret = FALSE;  }  return ret;  }  int APIENTRY WinMain(HINSTANCE hInstance,HINSTANCE prev,LPTSTR lpcmd,int cmdshow)  {  DialogBox(hInstance,MAKEINTRESOURCE(IDD\_DIALOG1),NULL,DlgProc);  return 0;  } |

### 效果：

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

### 注意：默认列表框是有排序功能的，我们的程序会使用编辑框由始指终都有输入焦点。

## 5.我们给对话框添加一个按钮文本为删除，ID为IDC\_BTN\_DEL，用来删除列表框下选中的项

|  |
| --- |
|  |

## 6.我们想实现的功能是：默认，我们把这个删除按钮设置为禁用，然后当我们选中一个列表框的项，就使得这个按钮可用，然后点击按钮会删除列表框指定是项，需要取捕获列表框发送的信息，实现启用和禁用按钮的代码如下：

|  |
| --- |
|  |

## 注意：LBN\_SELCHANGE消息处理不要和LBN\_KILLFOCUS消息处理同时使用，会有问题

## 7.然后我们添加点击删除按钮的事件处理代码

|  |
| --- |
|  |

### 效果：删除按钮一开始是禁用的当你选择了一个项后，它就启用了，需要选中一项才能删除

## 8.然后我们给对话框添加一个清空按钮，ID为IDC\_BTN\_CLEAR,然后我们在添加2个按钮，一个选中列表框的第一项，另外一个选中列表框的最后一项，ID:IDC\_BTN\_FIRST和IDC\_BTN\_LAST,为了好看一点，我们修改了一下布局

|  |
| --- |
|  |

## 9.然后我们给这个清空按钮添加点击事件处理代码

|  |
| --- |
|  |

## 10.给FIRST按钮添加点击事件处理代码

|  |
| --- |
|  |

## 11.给LAST按钮添加点击事件代码，这里需要一个变量count保存获取到的列表框有多少项，然后选中最后一项

|  |
| --- |
|  |

## ，然后我们来代码代码，注意列表框最后一项的所言是count-1，因为索引从0开始

|  |
| --- |
|  |

### 效果

|  |
| --- |
|  |
|  |

## 12.然后我们添加一个取消选择按钮，id为IDC\_BTN\_CLEAR\_SEL，点击一下会取消当前选中项，需要调整一下对话框的布局，如图

|  |
| --- |
|  |

## 13.然后我们来添加取消选中按钮的点击事件处理代码,注意是使用WM\_SETCURSEL消息，把wParam设置为-1

|  |
| --- |
|  |

### 效果

|  |
| --- |
|  |
|  |

# 这一节的学习到此为止，下一节我们继续学习更多功能，本节完整代码如下

## 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;  TCHAR szItem[128];  static int curSel;  static int count;    switch(uMsg)  {  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);  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;  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;  } |