Faculty of Computers, Informatics and Microelectronics

Technical University of Moldova

WINDOWS PROGRAMMING

Laboratory work #2

Advanced Form Elements. Child Windowses. Basics of Working With Keyboard.

Authors:
Irina Ungureanu

Supervisor:
Irina Cojanu

Laboratory work #2

1 Laboratory Work Requirements

– **Basic Level (grade 5 - 6) you should be able to:**

a) Create a Windows application what will dispaly a dialog box on some event (ex. on clicking some button)

b) Add a system menu to your application with at least 3 items (add actions to that items)

c) Hook keyboard input. Add 2 custom events for 2 different keyboard combinations (ex. change window background on ctrl+space)

– **Normal Level (grade 7 - 8) you should be able to:**

a) Realize the tasks from *Basic Level*.

b) Add a scroll bar that will change any visible parameter of any other element (color of a text) OR other 2 scroll bars that will manage main window size or position

– **Advanced Level (grade 9 - 10) you should be able to:**

a) Realize the tasks from *Normal Level*.

b) Customize your application by adding an icon and using different cursor in application

c) Add a listbox and attach some events when any element is accessed (clicked)

2 Laboratory work implementation

2.1 Tasks and Points

- Create a Windows application what will display a dialog box on some event

Firstly, I declared the dialog box in the resources file. There, I wrote what it should look like, what text it should contain and where it should be placed. Then, I created a dialog procedure to process messages for this box (clicking the ok button, for instance). After that, in the WinProc() function I called the DialogBox() to show the dialog box. It will appear when the Hotkeys menu tab is selected.

```
DialogBox(hInstance, MAKEINTRESOURCE(ID_DLGBOX), hwnd, DlgProc);
```

- Add a system menu to your application with at least 3 items

Firstly, I added the menu ID in the WINCLASSEX structure. Then, I declared the menu in the resources.rc file where I specified which tabs it should contain and the message ID that corresponds to the tab. Then, these messages are processed in the WM_COMMAND message, which adds functionality to the menu.

- Hook keyboard input. Add 2 custom events for 2 different keyboard combinations

In order to add hot keys to the application, first of all, I registered the key combination using the RegisterHotKey() function in the WM_CREATE message. Each hot key has a handle ID that are processed in the WM_HOTKEY message which specifies what will happen when the key combination is pressed.

```
RegisterHotKey(hwnd, HK_EXIT, MOD_CONTROL, 0x57); \\hotkey ctrl + Q
```

- Add a scroll bar that will change any visible parameter of any other element

There are 3 scroll bars in this application that represent and RGB color code. By moving these scroll bars the user can change the background color. For this, I declared the scroll bars in the WM_CREATE message. After that, in the WM_HSCROLL message I processed all the actions adherent to the scroll bar (ex: move one line left/ right or move one page left/right). In these messages, the current position of the thumb is incremented or decremented in accordance with the message. Then, I set the scroll position to the resulted value using the SetScrollPos() function.

Secondly, in order to change the background, I replaced the handle to the background brush according to the three values. And after that, in the WM_CTLCOLORSTATIC I set the background color using the SetBkColor() function.

```
SetScrollPos(scrollBar[i], SB_CTL, color[i], TRUE);
```

- Customize your application by adding an icon and using different cursor in application

For the icon, I declared in the resources the file directory of the icon and the handle corresponding to this icon. Then, in the WINCLASSEX structure, I loaded the icon in the hIcon and hIconSm fields.

```
wincl.hIcon = LoadIcon (hThisInstance, MAKEINTRESOURCE(ID_ICON));
wincl.hIconSm = LoadIcon (hThisInstance, MAKEINTRESOURCE(ID_ICON));
```

For the cursor, again, I declared it in the resources file. Then, in the WINCLASSEX structure I initialized the hCursor field using the icon handle. Then, in the WM_SETCUESOR message I set the new cursor using the SetCursor() function if it is located in the client zone, and if not, the cursor is set by default by the operating system.

```
SetCursor(LoadCursor(hInst, MAKEINTRESOURCE(ID_CURSOR)));
```

- Add a listbox and attach some events when any element is accessed

Firstly, I created the list box using the CreateWindowEx() function. Then, in the ID_LISTBOX message I check if any element of the listbox was double clicked and if yes, then the item is deleted using the fucntion:

```
SendMessage(listBox, LB_DELETESTRING, index, 0);
```

In order to add elements to the list box, I created a button that executes the following function:

```
SendMessage(listBox, LB_ADDSTRING, 0, (LPARAM)textStore);
```

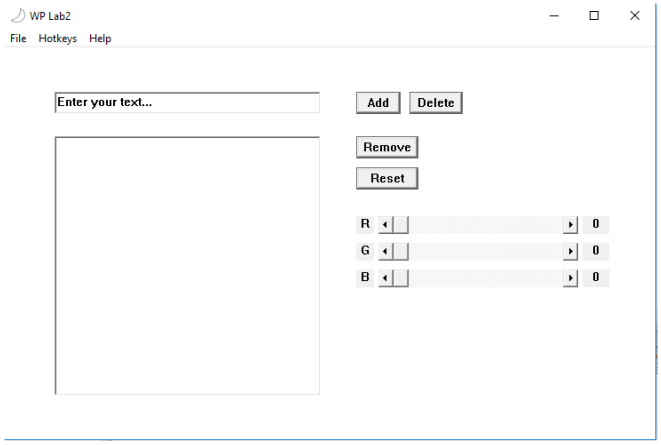
Then, in order to reset the whole list box the second parameter should be changed to LB_RESETCONT
On the other hand, in order to delete one single element the second parameter should be changed to LB_DELETESTRING and the third parameter should be the index of the element we want to delete. The index is calculated using:

```
index = SendMessage(listBox, LB_GETCURSEL, 0, 0);
```

2.2 Laboratory work analysis

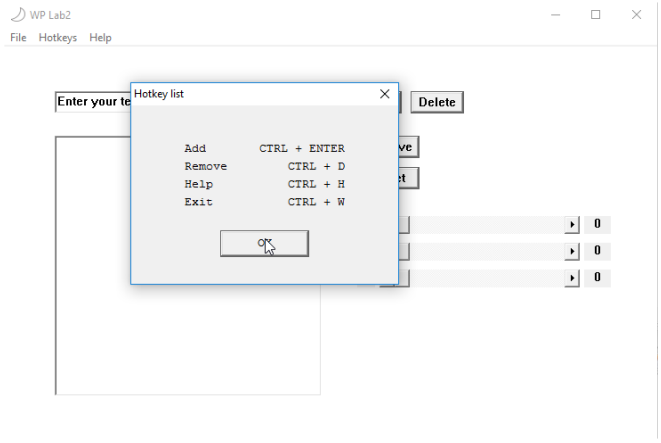
https://github.com/taurrielle/WP

2.3 Prove your work with screens

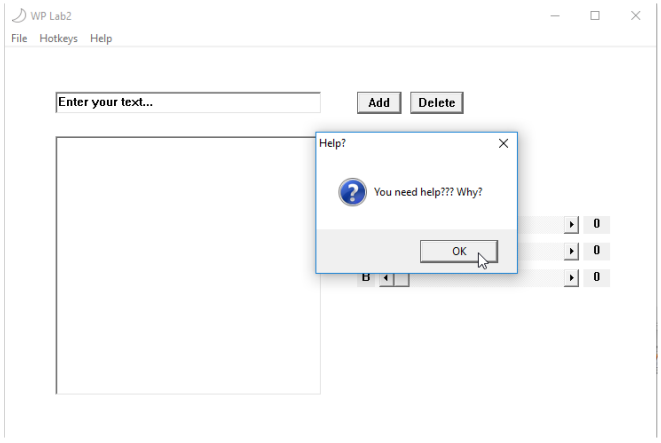


The basic window

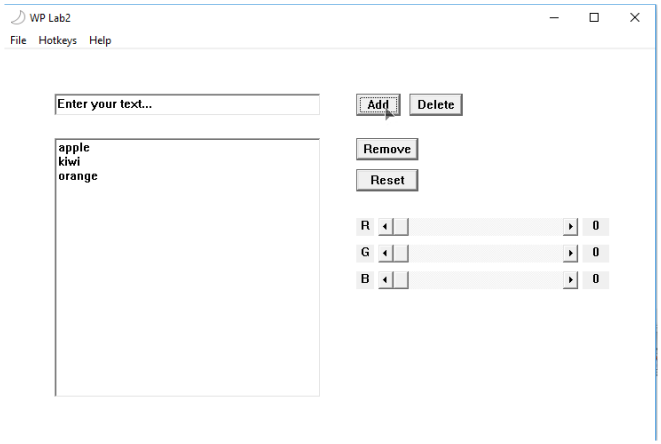
Selecting "About" from menu



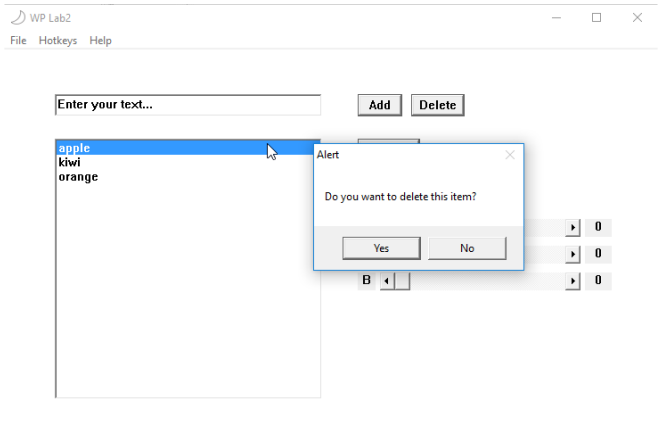
Selecting "Hotkeys" from menu



Selecting "Help" from menu

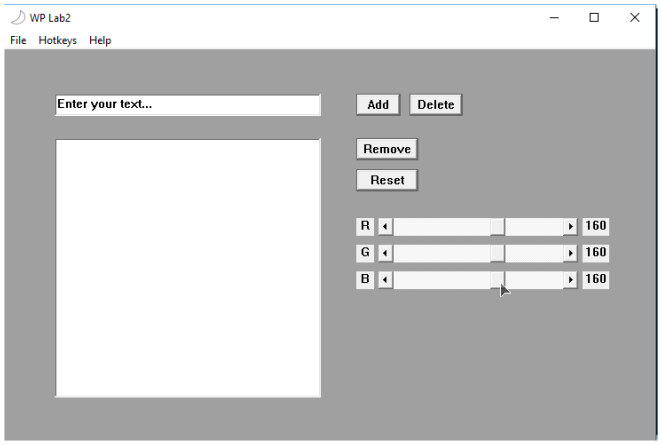


Clicking the Add button



Double clicking the list item

Clicking the Reset button



Changing the background using scroll bars

Conclusions

During this laboratory work, I learned about child windows and how to work with the keyboard. Also, I understood what a dialog box is, and how it is different from a simple message box. I created a window and added a list box that gets its items from a edit control. The items can be removed by double clicking, by pushing a button and by using the hot key ctrl + D. Also, this application has a menu from which the user can open two dialog boxes, one message box and close the application. The dialog boxes can also be opened using hot keys. Furthermore, the user can change the background color using three scroll bars that correspond to an RGB code. All in all, this laboratory work gave me insight regarding scroll bar functionality and including keyboard input in a Windows application.

References

1 Dialogs, `http://www.winprog.org/tutorial/dialogs.html`

2 Charles Petzold, *Programming Windows, 5th Edition*, 1998