Technical University of Moldova

Faculty of Computers, Informatics and Microelectronics

Software Engeneering and Automatics Department

Event-Driven Programming

Laboratory work #2

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

Author:
Vasile Drumea

Supervisor:
Mihai Coșleț

Chișinău 2018

Laboratory work #1

1 Purpose of the laboratory

Gain knowledge about advanced form elements, child windows and basic workflow with keyboard.

2 Laboratory Work Requirements

- **Mandatory Objectives**
  - Display a dialog box on some event (ex. on clicking some button)
  - Add a system menu to your application with at least 3 items (add actions to that items)
  - Add a scroll bar that will change any visible parameter of any other element (color of a text)
  - Hook keyboard input. Add 2 custom events for 2 different keyboard combinations (ex. change window background on ctrl+space)
- **Objectives With Points:**
  - `(2pt)` Add a listbox and attach some events when any element is accessed (clicked)
  - `(1pt)` Add 2 scroll bars that will manage main window size or position
  - `(1pt)`Customize your application by adding an icon and using different cursor in application
  - `(1pt)` Use a scroll bar to scroll through application working space. Scroll should appear only when necessary (eg. when window width is smaller than 300px)

3 Laboratory work implementation

3.1 Tasks and Points

In this laboratory work i implemented all needed tasks. The application was implemented in CodeBlocks IDE in C++ language using the template for a Win32 GUI application.

In what follows I will enumerate how i solved the tasks.

- **Mandatory Objectives**
  - When we press *delete* key a dialog box appears.
  - My system menu consists of 3 items : File ,Random and About. Random item contains 3 subitems. All the items have an associated action.
  - I've added a scroll bar that changes the background color of main window.
  - I've hooked 2 keyboard inputs. First is $ctrl + w$ to exit application and second is $ctrl + h$ for help.
- **Objectives With Points:**
  - `(2pt)` Added a listbox and an edit box with which we can add strings to listbox. If we right click a string we can delete it.
  - `(1pt)` Added scroll bars that will manages the main window.
  - `(1pt)` Customized my application by adding an icon and using different cursor in application.
  - `(1pt)` Added a scroll bar to scroll through application working space vertically when heigth is less than 300px.

3.2 Laboratory work analysis

In this laboratory work I used different types of child windows. In WM_CREATE I've created a listbox, a button, an editbox and a scrollbar. After theri creation I used 2 methods to set information about scrollbar :

```
SetScrollRange (scrollBar, SB_CTL, xMin, xMax, FALSE);
SetScrollPos (scrollBar, SB_CTL, pos, TRUE);
```

Also, after this I registeret 2 hotkeys with $RegisterHotKey()$ method. This are for $ctrl + w$ and $ctrl + h$ combinations. The first one closes the application and the second envokes a messagebox.

To manage the resources I use 2 files "macros.h" where i keap all the macros and "resource.rc" where are resources i.e. the icon, dialogbox and the menu.

The icon is keapt in additional .ico file.

The dialog box appears when we press "delete" key. It is a simple dialog box with a title, a static text and 2 buttons which close the dialogbox.

The menu has 3 items. In file item we have the exit option. In random file there are 2 options, first one to move the window to a random place and the second one to change background color to a random one. Also there is a grayed option. The last item, About envokes a messagebox.

Icon and menu are introduced to the app in WinMain with the 2 methods :

```
LoadIcon (GetModuleHandle (NULL), MAKEINTRESOURCE (IDI_ICON));
MAKEINTRESOURCE(IDM_MENU);
```

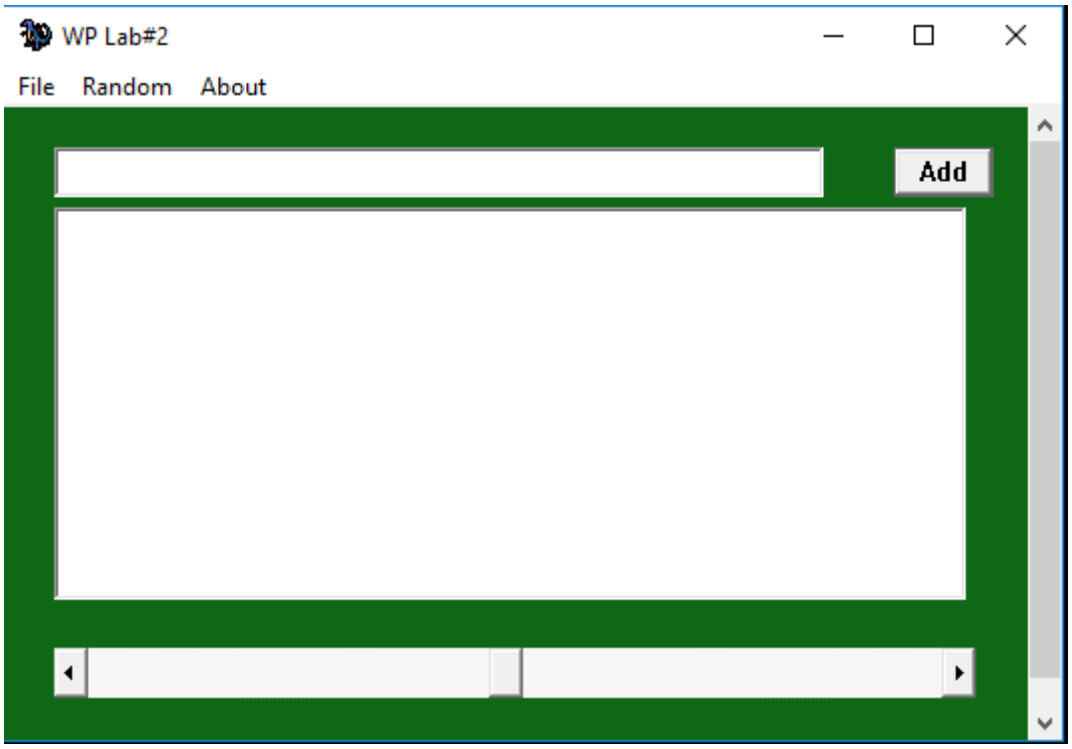
The listbox stores string addes from the editbox with the add button. If we want to remove an item we do it with righclick on a highlighted item. This function is added at $WM\_CONTEXTMENU$ message:

```
if (MessageBox (hwnd, TEXT ("Remove selected item?"),
    TEXT ("Warning"), MB_YESNO) == IDYES) {
        int id = SendMessage (GetDlgItem (hwnd, IDC_LISTBOX),
                                                LB_GETCURSEL, 0, 0);
        SendMessage (GetDlgItem (hwnd, IDC_LISTBOX),
                        LB_DELETESTRING, id, 0);
}
```

The scrollbar is used to change the background color of the main window to a random one. Its action is treated in case of $WM\_HSCROLL$. For the other scroll bar which is a vertical one, I treated the $WM\_VSCROLL$. The vertical scrollbar appears when the heigth is $< 320px$.

Here is a link to my repository for Event-Driven Programming laboratory number 2 :
https://github.com/Wazea/WP/tree/master/lab-2

3.3 Prove of the work with screens



Conclusion

- In this laboratory work i gained knowledge about child windows, basic workflow with keyboard and the main concepts about scrollbars.
- The execution shows that the program works as expected and the tasks were completed.
- All the keyboard inputs are encoded as virtual keys. Also we can use the keys in combination with alt and ctrl keys or other keys by indicating $MOD\_SOMEKEY$ parameter to RegisterHotKey() method.
- It is recommended to include the instructions in the case acording to the message with which they are related. It is a bad practise to use same messages in different message handler because it makes the Window Procedure more chaotic.
- It is possible to send artificially a message with SendMessage().

References

1 MSDN, *Official page*, https://msdn.microsoft.com/en-us/library/windows/desktop/ff657751(v=vs.85).aspx

2 The First Reading, *Landing page*, http://www.winprog.org/tutorial/start.html