Faculty of Computers, Informatics and Microelectronics

Technical University of Moldova

WINDOWS PROGRAMMING

Laboratory work #1

Window. Basic window's form elements

Authors:
Daniel Surdu

Supervisor:
Irina Cojanu

Laboratory work #2

1 Purpose of the laboratory

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

2 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)

3 Laboratory work implementation

3.1 Tasks and Points

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

First of all I created the menu of the window, there I put the button info. When you press on that button on screen appears the dialogbox which have in in the information of the author of the program (me)

For this I should create another procedure which is pretty similar with WindowProcedure() and I named it DialogProc(). With DialogProc() function you can work the same as with WindowProcedure(). To create the visual part of the dialogbox I had to create another file with extension .rc, and for dialogbox code looks like that:

```
IDM_DLGBOX DIALOG 100, 50, 160, 100
STYLE DS_MODALFRAME | WS_POPUP | WS_CAPTION | WS_SYSMENU
CAPTION "Info"
FONT 8, "MS Shell Dlg"
BEGIN
    DEFPUSHBUTTON   "OK", IDOK, 55, 80, 50, 14
    CTEXT   "Program made by", IDM_STATIC, 0, 20, 160, 40
    CTEXT   " Unnemotional Hyena", IDM_STATIC, 0, 30, 160, 40
END
```

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

To add a system menu to my application I declared a default IDM_MENU and I wrote in the resources.rc:

```
IDM_MENU MENU DISCARDABLE
BEGIN
    POPUP "&File"
    BEGIN
        MENUITEM "&This",   IDM_FILE_THIS
        MENUITEM "&Close",  IDM_FILE_CLOSE
    END
    POPUP "Help"
    BEGIN
        MENUITEM "Info",    IDM_FILE_INFO
    END
END
```

To add this menu to my window I simply added MAKEINTRESOURCE(IDM_MENU) to the wincl.lpszMenuName.

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

To add 2 custom events for 2 different keyboard combinations I simply had to register 2 hotkeys:

```
RegisterHotKey(hwnd, IDM_EXIT_KEY, MOD_CONTROL, 0x57);
RegisterHotKey(hwnd, IDM_MAXIMIZE_KEY, MOD_SHIFT, VK_RETURN);
```

Where first parameter is handle to the window, second is the unique ID defined in header, third is first key in the combination (MOD_CONTROL is Ctrl key, 0x57 virtual code of the "w", MOD_SHIFT is shift button and VK_RETURN is ENTER button)

- 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

I added two scroll bars that change size of the window, one scroll bar change the x coordinate and another y. To do this I created 2 scroll bars using CreateWindowEx() function with second parameter named (TEXT("SCROLLBAR")). To change the position of the Scroll I used SetScrollPos() function and I made it move in the WM_HSCROLL message where I write all the movements needed. To change the scroll position when you resize manually the window I went into WM_SIZE message and intercept the changes made in window.

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

Adding a icon is literally super simple. You put the .ico file in program folder, write the resource.rc (IDM_ICON ICON DISCARDABLE "icon.ico") add an ID to that icon in header and add in window class:

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

To add a different cursor you need to add .cur file in the program folder to write in resurce.rc (IDM_CURSOR CURSOR DISCARDABLE "cursor.cur") add an ID in the header file write in the winclass (wincl.hCursor = LoadCursor (NULL, IDC_ARROW);) and you have to add in WM_SETCURSOR message where and what cursor to appear.

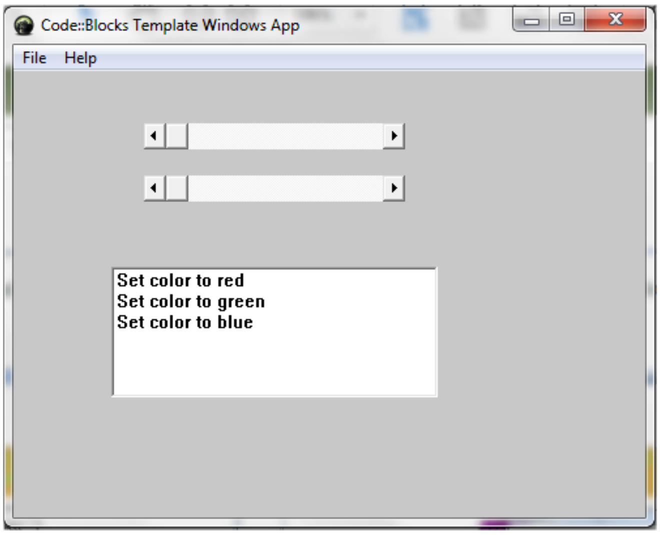
- Add a listbox and attach some events when any element is accessed (clicked)

Using CreateWindowEx() with second parameter TEXT("LISTBOX") I created the list box. I used SendMessage() to add the 3 text lines. After that I go into case IDM_LISTBOX and check if is double clicked on any of the text lines, and when one is pressed it creates a brush with the color I need and in message WM_CTLCOLOSTATIC I draw with that brush the background of the client zone.

3.2 Laboratory work analysis

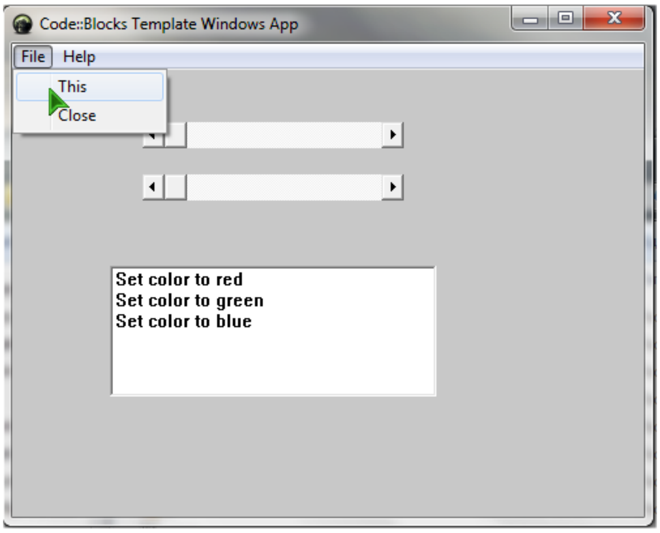
https://github.com/UnnemotionalHyena/WP

3.3 Prove your work with screens



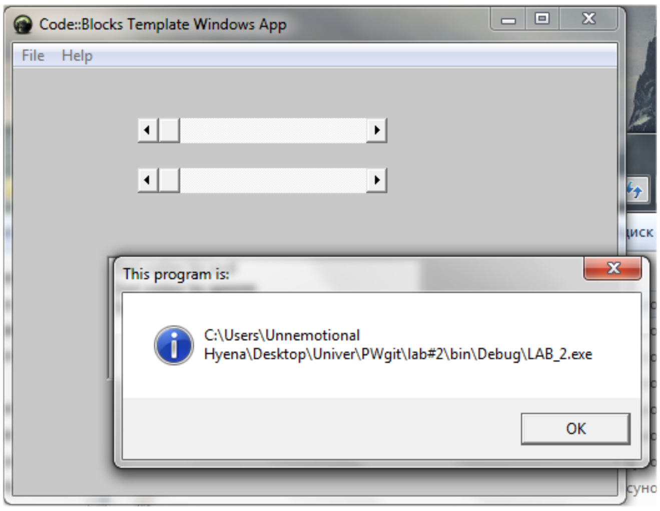
Set color to red
Set color to green
Set color to blue

The basic window



Set color to red
Set color to green
Set color to blue

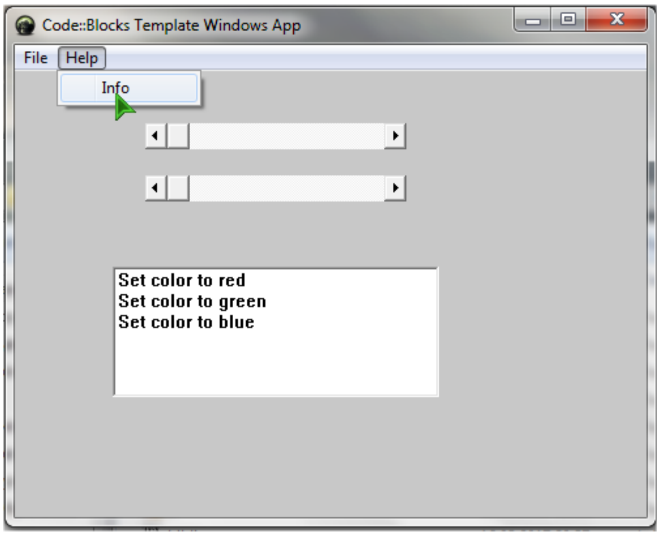
Clicking the File button



This program is:

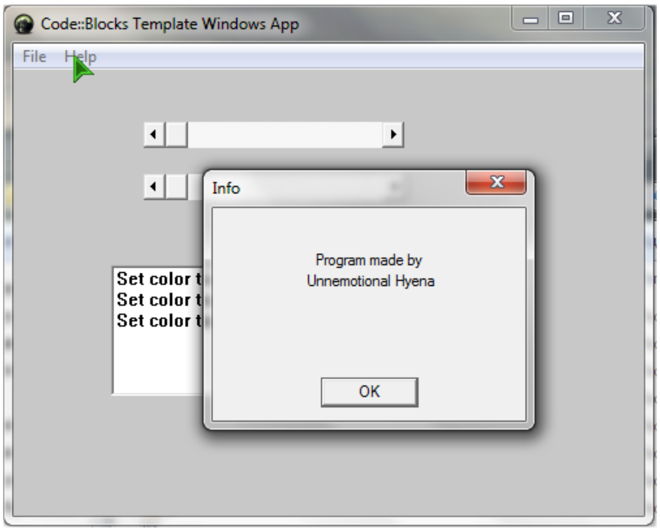
C:\Users\Unnemotional Hyena\Desktop\Univer\PWgit\lab#2\bin\Debug\LAB_2.exe

Clicking the This button



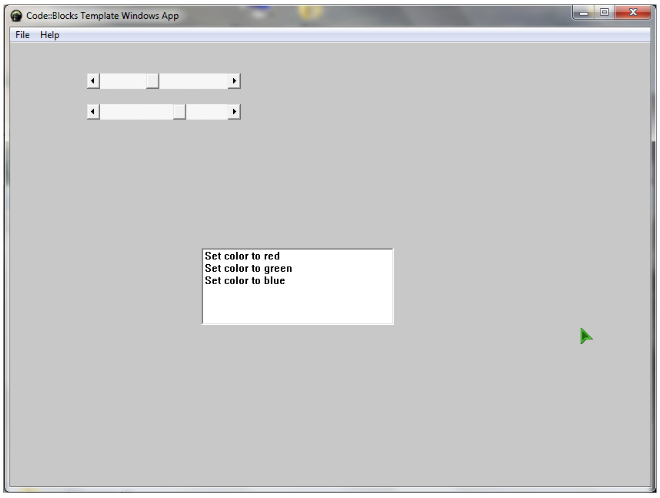
Set color to red
Set color to green
Set color to blue

Clicking the Help button

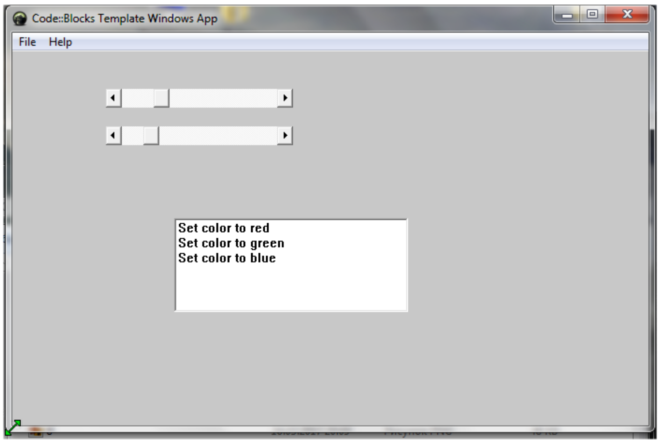


Program made by
Unnemotional Hyena

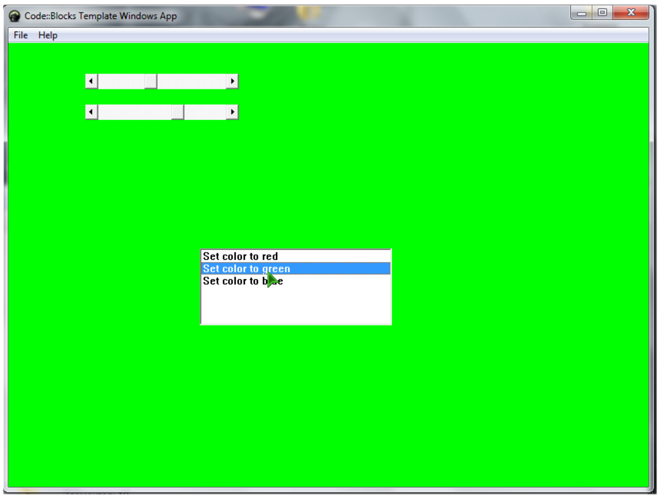
Clicking the Info button



Use scroll bars



Resize manually



Double click on ”Set color to green”

Conclusions

During this laboratory work I learned how to make a window menu, how to work with resurce.rc, how to work with scrolls and basic of working with keyboard. Everything was made using win32 api and C++ language.

References

1 Charles Petzold, *Programming Windows, Fifth Edition*, 1998

2 MSDN, `https://msdn.microsoft.com/`

3 Github, `https://github.com/`