Faculty of Computers, Informatics and Microelectronics

Technical University of Moldova

WINDOWS PROGRAMMING

Laboratory work #1

Window. Basic window's form elements

Authors:
Irina Ungureanu

Supervisor:
Irina Cojanu

Laboratory work #1

1 Purpose of the laboratory

Gain knowledge about basics of event-driven programming, understanding of window's class and basic possibilities of Win32 API. Also she will try to understand and process OS messages.

2 Laboratory Work Requirements

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

a) Create a Windows application

b) In the middle of the window should be present the following text: "Done with Pride and Prejudice by student name". Replace student name with your name.

c) On windows resize, text should reflow and be in window's middle (vertically and horizontally)

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

a) Realize the tasks from *Basic Level*.

b) Add 2 buttons to window: one with default styles, one with custom styles (size, background, text color, font family/size)

c) Add 2 text elements to window: one with default styles, one with custom styles (size, background, text color, font family/size)

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

a) Realize the tasks from *Normal Level*.

b) Make elements to interact or change other elements (2 different interactions) (ex. on button click, change text element color or position)

c) Change behavior of different window actions (at least 3). For ex.: on clicking close button, move window to a random location on display working space

3 Laboratory work implementation

3.1 Tasks and Points

- Create Windows application

In order to create a Windows application, open an IDE (I used CodeBlocks) and go to File → New → Project → Win32 GUI Project

- Put text in the middle of the window

For this, I used the DrawText() function inside the WM_ PAINT message in the following way:

```
DrawText(hdc, TEXT ("HELLO"), -1, & rect, DT_SINGLELINE | DT_CENTER | DT_VCENTER);
```

This function displays the text "HELLO" in the center of the client area by using the DT_ CENTER and DT_ VCENTER identifiers. The first argument is a handle to the device context returned from BeginPaint(). The second argument is the text to draw, the third argument is set to -1 to indicate that the text string is terminated with a zero character and the forth is a pointer to the client rectangle. Also, on window resize it will remain in the middle because or window class style i used the CS_ HREDRAW and CS_ VREDRAW flags.

- Add buttons: with default styles and with custom styles

In order to create a button with default styles, the function CreateWindowEx() with the second parameter set to "BUTTON" was used. For creating one with custom styles we need to do the following. Firstly, a BS_ OWNERDRAW flag should be introduced in the function. Secondly, in the WM_ DRAWITEM message I set the text color and the background color using the SetTextColor and SetBkColor respectively. Also, the ExtTextOut function puts the text on the button and the DrawEdge function draws the edge around the button. Finally, I changed the text font using the SendMessage function with which I have sent the previously defined font using the CreateFont() function to the button object.

```
SendMessage(button3, WM_SETFONT, (WPARAM)hfFont, MAKELPARAM(FALSE,0));
```

- Add text elements: with default styles and with custom styles

In the same way as the buttons, the text element is created using the CreateWindowEx() function with the second parameter "EDIT". In order to change the text font I defined a custom font using CreateFont() and sent it with SendMessage(). Furthermore, I changed the text color and background color using the SetTextColor() and SetBkColor() in the WM_ CTLCOLOREDIT message.

- Add combobox

There is a combobox for choosing the text color in this application. I created it using CreateWindowEx() with the second parameter "COMBOBOX". I added the items in the list using SendMessage():

```
SendMessage(comboBox, CB_ADDSTRING, 0, (LPARAM)"Select color");
```

Moreover, we know when an item is selected using a bunch of if statements in the ID_ COMBOBOX message. Every if sets a flag to true which is then processed in the WM_ CTLCOLORSTATIC message and changes the text color with the SetTextColor() function according to the flag.

- Make elements interact or change other elements

Firstly, when the "OK" button is clicked the text in the first two text areas is concatenated a sent to the third text area using the SendMesage() function.

Secondly, when the "Reset" button is clicked an empty text is sent to the third text area and the default texts are sent to the first and second areas, again, using the SendMessage() function.

Furthermore, the "Change font" button changes randomly the font in text area 3, by choosing and creating a random font from an array and sending it to the text object with SendMessage().

Finally, when the "Click!!!" button is clicked the text from text area 3 is copied and passed into a message box to create a custom message. For this I used the MessageBox() function.

```
MessageBox(NULL, TEXT(result), TEXT("Hello"), MB_OK | MB_ICONWARNING);
```

- Change behavior of different window actions

I changed the behavior of the three standard right top buttons. In the WM_ SYSCOMMAND I considered three cases: SC_ MINIMIZE, SC_ MAXIMIZE, SC_ CLOSE. For the minimization button I created a function that checks if the Google Chrome browser is opened and opens a message box with a text appropriate for the situation. The maximization button has the fuction of moving the window to a random position using the SetWindowPos() function and two randomly generated variables for x and y positions.

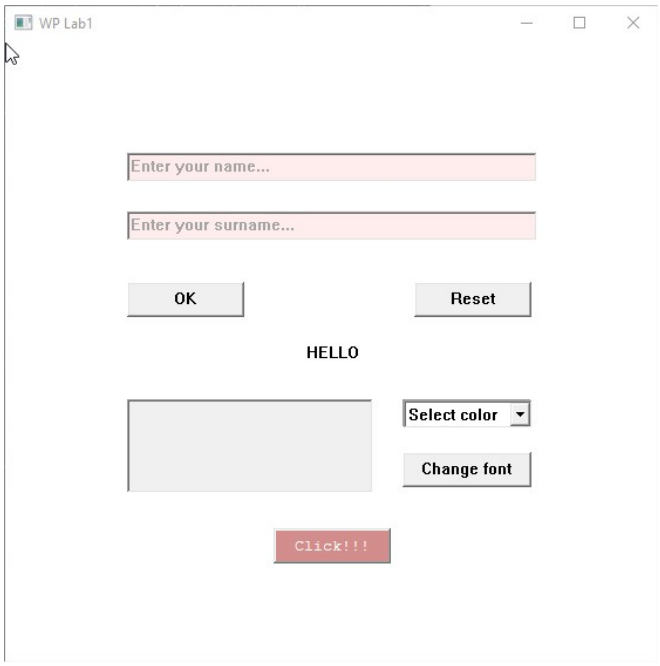
```
SetWindowPos( hwnd, 0, xPos, yPos, 0, 0, SWP_NOZORDER | SWP_NOSIZE );
```

The close button just creates a message box that asks the user if he really wants to quit the application.

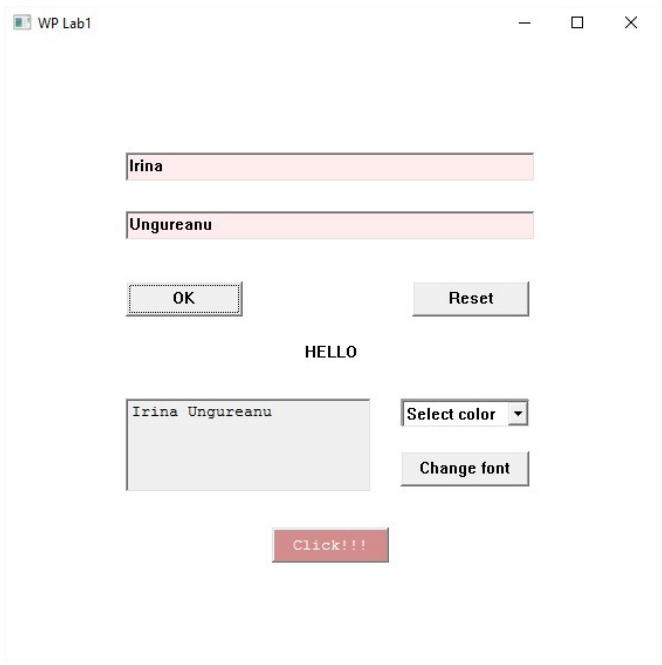
3.2 Laboratory work analysis

https://github.com/taurrielle/WP

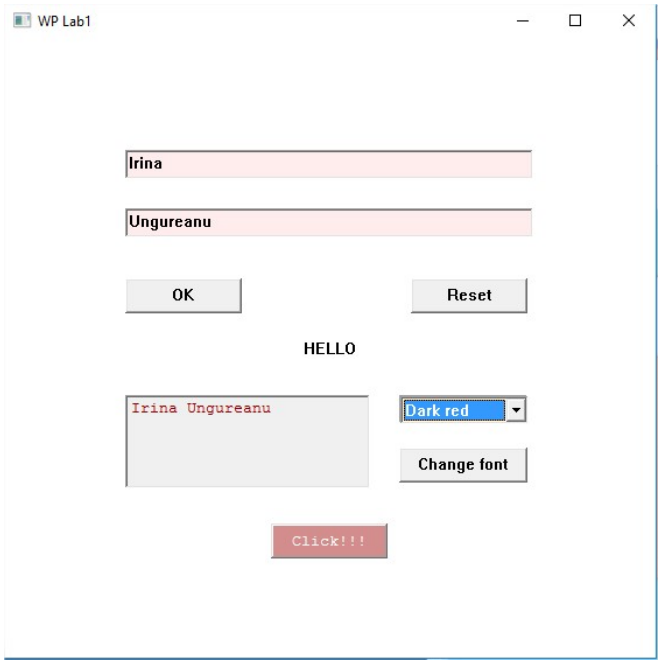
3.3 Prove your work with screens



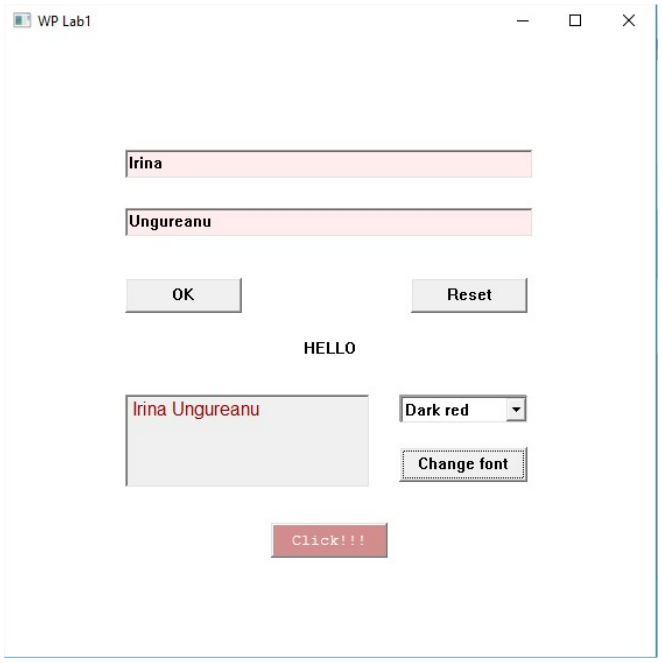
The basic window



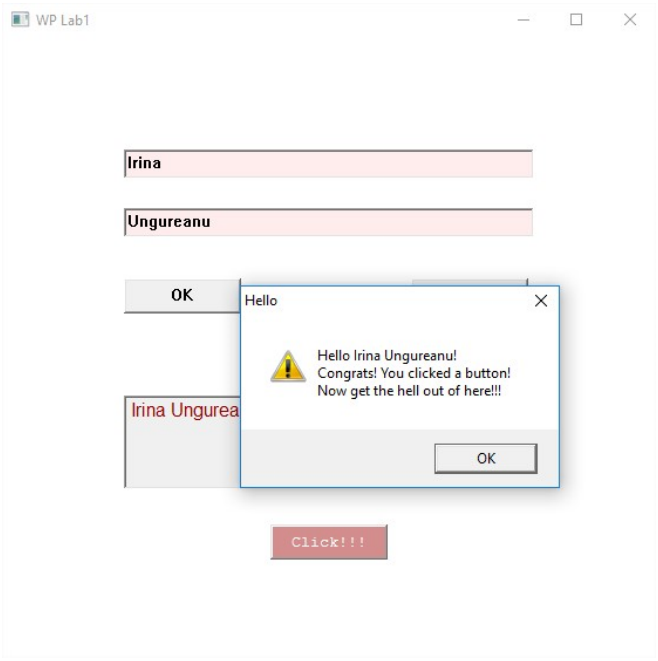
Clicking the OK button



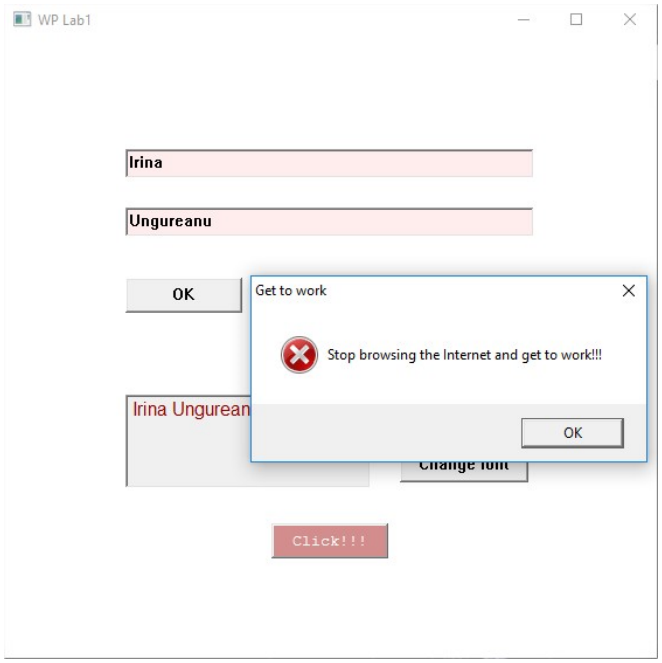
Changing the text color to red



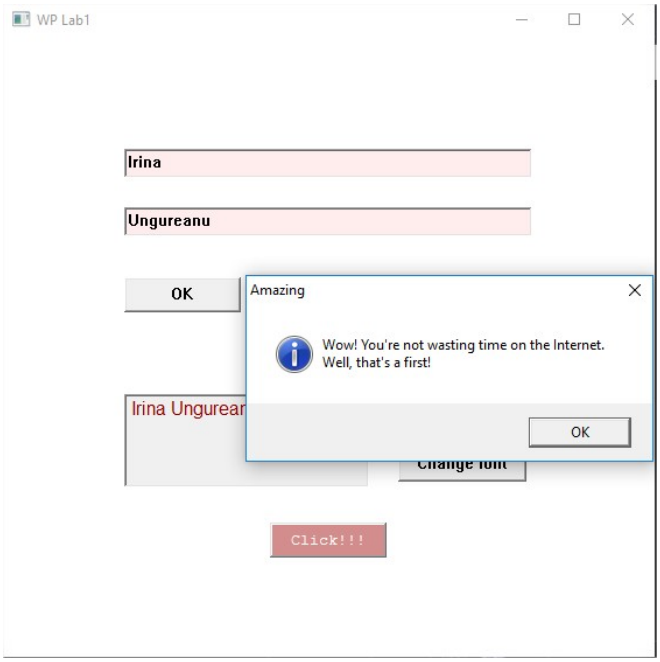
Changing the font



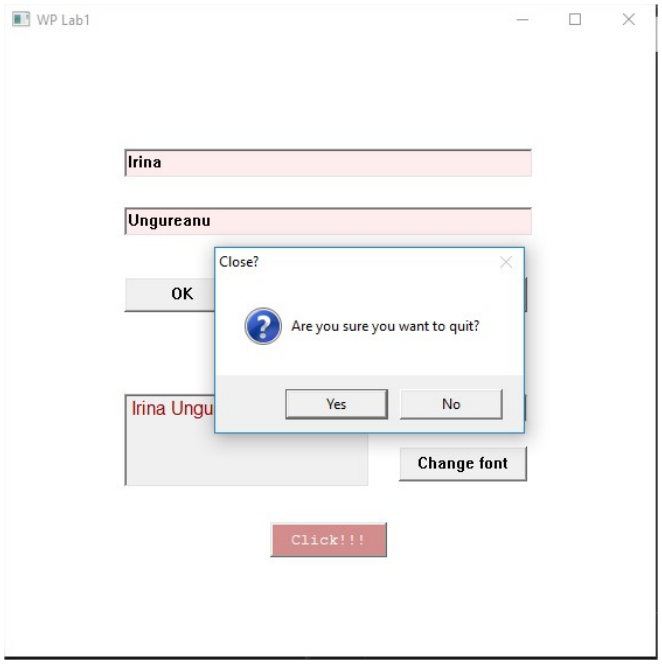
Clicking the CLICK!!! button



Clicking minimize when browser open



Clicking minimize when browser closed



Clicking the close button

Conclusions

During this laboratory work, I learned the basics of Windows Programming. I understood how the message processing in Windows applications works and got to use the basic functions included in the windows.h library. I created a window and added some buttons, text fields and even overwritten some default buttons (like the close button or the minimize button). I learned how to make a custom button and a custom text area, how to change the font, text color and background. Moreover, I learned how to add functionality to the buttons and make them change other elements (for example, to change the color of the text in a text area). All in all, this laboratory work gave me insight regarding event-driven programming, Windows applications and how to work with them.

References

1 Windows Programming Basics, `www.winprog.org/tutorial/start.html`

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