Faculty of Computers, Informatics and Microelectronics

Technical University of Moldova

WINDOWS PROGRAMMING

Laboratory work #3

Basics of Working with Mouse. GDI Primitives. Bezier Curve.

Authors:
Irina Ungureanu

Supervisor:
Irina Cojanu

Laboratory work #3

1 Laboratory Work Requirements

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

a) Draw 5 lines of different colors and weights

b) Draw 2 Bezier curves

c) Draw 4 plane objects (ex. circle, square, pie, polygon...) of different colors, weights, filled and not

d) Draw 2 different objects using mouse

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

a) Realize the tasks from *Basic Level*.

b) Draw a custom bitmap image

c) Fill 2 object with gradient

d) Hook keyboard input. Add 2 different keyboard combinations that will change mouse ability to draw objects (ex. on Ctrl+C will draw circles, on Alt+R will continue to draw circles but of read color)

e) Draw a Bezier curve using mouse

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

a) Realize the tasks from *Normal Level*.

b) Zoom in and out application working area using keyboard or mouse wheel

c) Use mouse as an eraser (choose 1 option):

1) delete objects using mouse clicking

2) eraser of a fixed width

3) eraser with adjustable width

– **for Bonus Point Tasks :**

a) Realize the task with mouse eraser for all 3 cases listed above. In order to choose one of them, add 3 buttons/icons or check boxes.

2 Laboratory work implementation

2.1 Tasks and Points

- Draw 5 lines of different colors and weights

In order to accomplish this, I firstly created a pen using the CreatePen() function. In the same function I specified the weight of the future line and its color. Then, I selected the pen and moved the cursor to the position where the line had to start. And called the function LineTo() in which I specified the coordinates of the point where the line should end.

```
HPEN linePen = CreatePen(PS_SOLID, 2, RGB(0, 0, 0));
hpen = (HPEN)SelectObject(hdc, linePen);
MoveToEx(hdc, 25, 40, NULL);
LineTo(hdc, 75, 40);
```

- Draw 2 Bezier curves

Firstly, I initialized an array of 4 points in accordance to which the Bezier curve will be constructed. Then, in the WM_PAINT message, I created the pen with which the curve would be drawn, selected it and called the function PolyBezier() where I passed as a parameter the array of points I declared earlier.

```
PolyBezier(hdc, Pt, 4);
```

- Draw 4 plane objects (ex. circle, square, pie, polygon...) of different colors, weights, filled and not

In this application, I drew a rectangle, an ellipse, a pie and a polygon. And, as before, I created a pen and selected it in the WM_PAINT message. For the rectangle I used the Rectangle() function where I specified the coordinates of the top left and bottom right corner.

```
Rectangle(hdc, 25, 180, 80, 210);
```

For the ellipse, I used the Ellipse() function with the same parameters as for the Rectangle() function. For the pie shape, I called the Pie function which take as parameters the top left corner and the bottom right corner of the imaginary rectangle that encloses the shape and the next 4 parameters are the points in the imaginary rectangle where the slices should be made.

```
Ellipse(hdc, 25, 270, 80, 300);
Pie(hdc, 30, 220, 80, 260, 80, 220, 80, 260);
```

Finally, for the polygon I just called the Polygon() function passing as a parameter an array of points through which the polygon should pass.

```
Polygon(hdc, polyPt, 4);
```

- Draw 2 different objects using mouse

In order to do this, I created two buttons: Rectangle and Ellipse. Then, if the left mouse button is pressed (WM_LBUTTONDOWN), I take the mouse coordinates, check if these coordinates are within the drawing area and put them in a RECT variable. The position of the mouse will be the upper left corner of the rectangle or ellipse. When the left mouse button is released, I capture the mouse coordinates and put them in the RECT variable. Then, I just create a pen, select it and call the Rectangle() or Ellipse() function, depending on the pressed button.

- Draw a custom bitmap image

For the bitmap image, I used the LoadImage() function to load it. Then, selected its handle and got its height and width using GetObject(). After which, drew it bit by bit using the function BitBlt()

```
SelectObject(hdcMem, hbmpImage);
GetObject(hbmpImage, sizeof(bitmap), &bitmap);
BitBlt(hdc, 0, 430, bitmap.bmWidth, bitmap.bmHeight, hdcMem, 0, 0, SRCCOPY);
```

- Fill 2 object with gradient

Firstly, I created a temporary rectangle variable and gave it the coordinates of the top left corner. Then, initialized the start and end colors of the gradient using 3 integer values. After that, in a for loop, that will be executed a number of times that is equal to the height of the rectangle, the color is calculated using the start and stop colors declared earlier and a rectangle with height 1 pixel is drawn using this color.

- Hook keyboard input. Add 2 different keyboard combinations that will change mouse ability to draw objects

This application has 2 hot key combinations: ctrl + R draws rectangles and ctrl + L draws lines. In order to accomplish this I registered the hotkeys in the WM_CREATE message. Then, in the WM_HOTKEY case, I set the flags that trigger rectangle and line drawing to true for the respective hotkeys. These flags are checked during the LBUTTONDOWN adn LBUTTONUP messages which enables the user to draw using mouse.

```
RegisterHotKey(hwnd, HK\_RECT, MOD\_CONTROL, 0x52); //ctrl+R
RegisterHotKey(hwnd, HK\_LINE, MOD\_CONTROL, 0x4C); //ctrl+R
```

- Draw a Bezier curve using mouse

Firstly, I declared an array of 4 variables of POINT type. Then, I catch the coordinates of the first point when the let mouse button is pressed and the second when this button is released. The third and the forth points are caught when the right mouse button is pressed and released. Then ,using these coordinates as a parameter I call the PolyBezier() function which draws the curve.

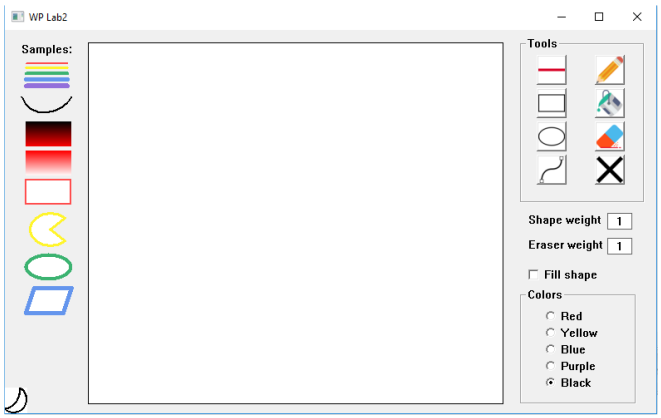
- Use mouse as an eraser

In order to create an eraser I used the idea of a white pencil. When the left mouse button is pressed I get the coordinates of the starting point. Then, as the cursor is moved through the drawing area, I just construct lines to consecutive points indicated by the mouse position. Also, I check if the cursor is within the drawing area in order not to erase something that is not supposed to be erased, like the client area background.

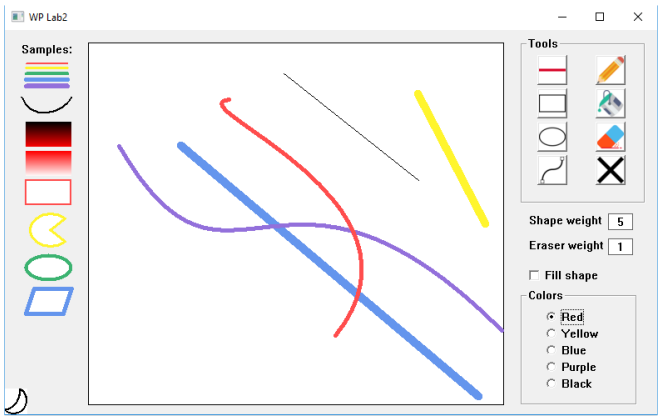
2.2 Laboratory work analysis

https://github.com/taurrielle/WP

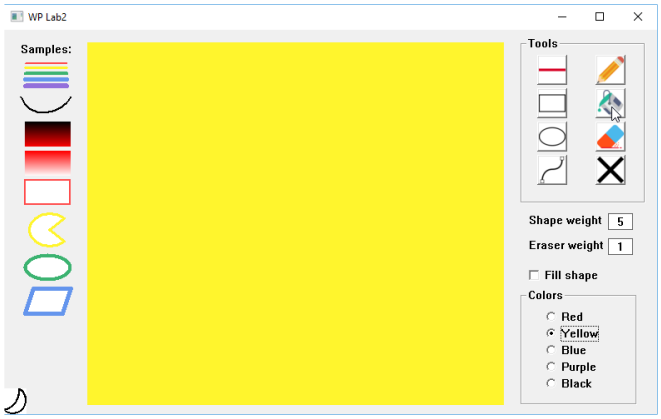
2.3 Prove your work with screens



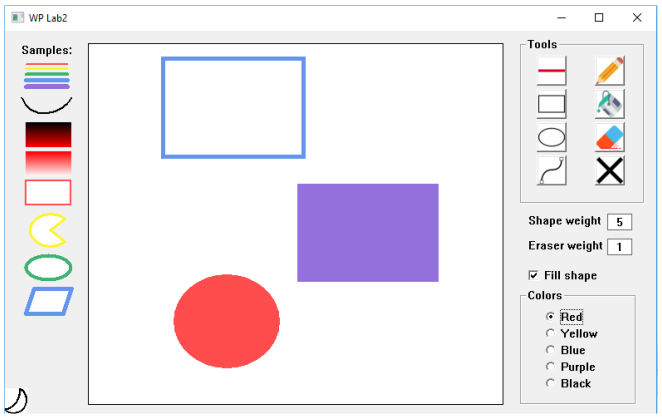
The basic window



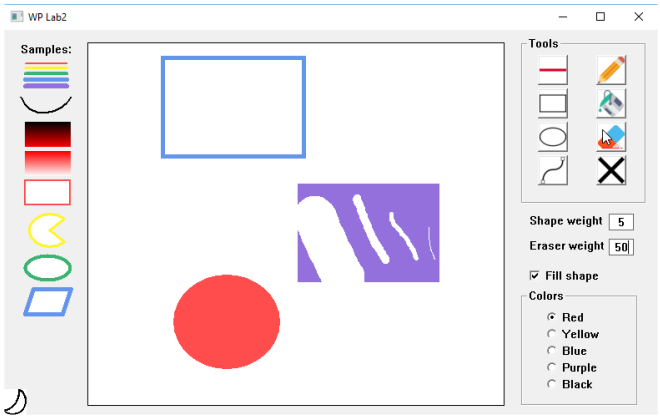
Drawing lines and Bezier curves



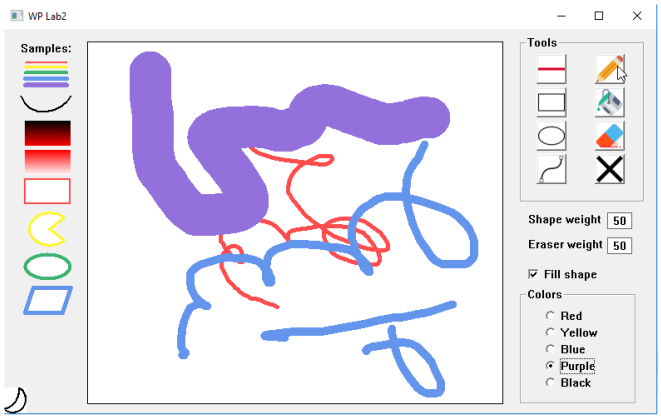
Clicking the "Fill" button



Drawing rectangles and ellipses



Using eraser



Using the ”Pen” tool

Conclusions

During this laboratory work, I learned about how to work with GDI. In my application, I drew some shapes of different colors and line weight. Also, I learned the logic behind Bezier curves and drawing gradients. Moreover, I found out how to implement mouse use in an application and which cases are used to process mouse moves and clicks. This enabled me to create an application resembling Paint. It has the possibility to draw several objects (rectangle, ellipse, line, bezier curves etc.) to choose the color and whether the object will be filled or not. All in all, this laboratory work got me to better understand how to work with GDI.

References

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