Technical University of Moldova

Faculty of Computers, Informatics and Microelectronics

Software Engeneering and Automatics Department

Event-Driven Programming

Laboratory work #4

Windows Timer. Animation.

Author:
Vasile Drumea

Supervisor:
Mihai Coșleț

Chișinău 2018

Laboratory work #4

1 Purpose of the laboratory

Gain knowledge about basics of working timer, its usage and how to implement animations.

2 Laboratory Work Requirements

- **Mandatory Objectives**
  - Create an animation based on Windows timer which involves at least 5 different drawn objects;
- **Objectives With Points:**
  - (2 pt) Increase and decrease animation speed using mouse wheel;
  - (2 pt) Solve flickering problem, please describe in your readme/report how you did it;
  - Add animated objects which interact with each other (2-6 pt), ex.:
    - Few balls which have different velocity and moving angles. In order to get max points, add balls with mouse, make balls to change color on interaction and any other things that will show your engineering spirit;
    - Any other interesting and reach in animation application;
  - Animate a Nyan Cat that leaves a rainbow tail (Math.floor(+35% for task with interacting objects))

3 Laboratory work implementation

3.1 Tasks and Points

- Create an animation based on Windows timer which involves at least 5 different drawn objects - My animation can involve multiple elements(how much the memory permits us to create);
- (2 pt) Increase and decrease animation speed using mouse wheel - The speed of animation is slowed down if the mouse wheel is rolled back and fasten up vice-versa;
- (0 pt) Solve flickering problem, please describe in your readme/report how you did it;
- Add animated objects which interact with each other (2-6 pt) - My animation with the balls contains interaction at the collision as explained in Laboratory work analysis;
- Animate a Nyan Cat that leaves a rainbow tail - For Nyan Cat animation press Ctrl + N;
- Total - About 8 pt + a bonus for Nyan Cat;

3.2 Laboratory work analysis

The first mandatory task was implemented by creating a bitmap which will contain the whole window. It is drawn initially with WHITE_BRUSH.

Also for the ball animation purpose I've created a Ball class. This class's properties are the color, center coordinates and the horizontal and vertical velocities. The methods of the class are the ones described in declarations.h. These are for changing the properties of an object. Also 2 methods to solve the collisions with the walls and between balls and a method to redraw each ball.

To solve the collisions I've used the formulas from [3]. You can observe the collision between the ball if the speed is not that high. Mostly the collision will be observed only by the change of colors of the balls. I couln't grant a perfect collision between the balls, because with win32 API I should've used the Ellipse method which takes int parameters, but for a more precise collision I would've used doubles.

As i previously said the balls at the collision change their color to a random one. The radius of each ball is 25px. You can add as many balls as you want because I've used vector class to store them. The balls are added at left mouse button click.

An example result can be seen in Figure 3.1 in screens section.

For the Nyan Cat animation I have 7 bmp images. If you pres the hotkey combination Ctrl + N my bool variable drawNyan enables WM_PAINT to draw it. This is done with BitBlt() method for succesive images.

RECOMENDATION :

If you want to see more accurately the Nyan Animation initially decrease the speed with mouse wheel and then start it.

Sorry if the later animation is not so eye-catching. It can be improved with better samples of frames. My are just cropped in paint with my poor precision.

Here is a link to my repository for Event-Driven Programming laboratories : https://github.com/Wazea/WP/tree/master/lab-4

3.3 Prove your work with screens

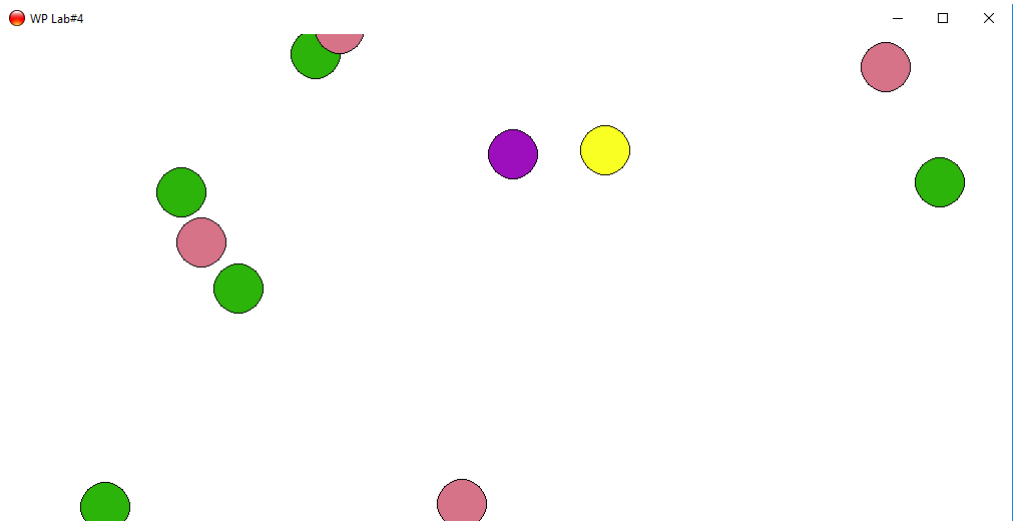

Figure 3.1– Balls Animation

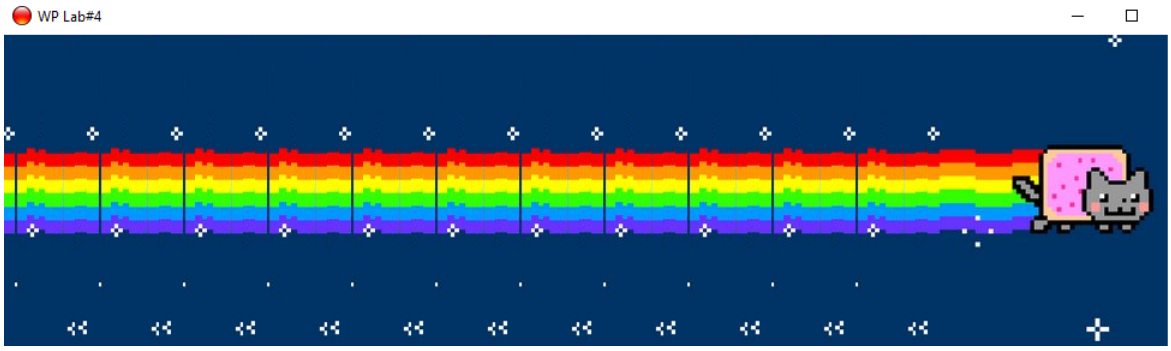

Figure 3.2– NyanCat Animation

Conclusion

– In this laboratory work i gained knowledge about Win32 app developement. The workflow with timer and some GDI functions.

– The execution shows that the program works as expected and the tasks were completed.

– A good PSG is crucial in app developement to offer readabillity and to maintain the order in the code.

– 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.

– If the speed of animation increases there appear irregularities in animation.

References

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

2 Programming Windows by Charlez Petzold, 5th edition, SectionI,Chapter8

3 About Elastic Collisions, https://en.wikipedia.org/wiki/Elastic_collision