Rendering Engine Development for the Game – OOP

Purpose of work

Implement the rendering engine component, which contains the rendering logic of the game.

Task

In this assignment, you are going to implement the rendering engine for the game you have chosen previously in Assignment #4.

Task milestones

To complete this assignment, it is highly recommended to use the SFML library. As an alternative Qt can be used. There are plenty of tutorials about SFML usage and the first steps of working with it. You can find these tutorials here.

!!! Please watch lectures that explain the basics of SMFL library setup and usage, also please reach your lecturer to get help in the environment setup. !!!

In order to visualize the game objects, it is better to use the primitives first. For example, if the player controls a tank or a spaceship in your game, you may use primitives like squares or rectangles to represent it. Later, you can also use more sophisticated forms (or even textures), for additional points.

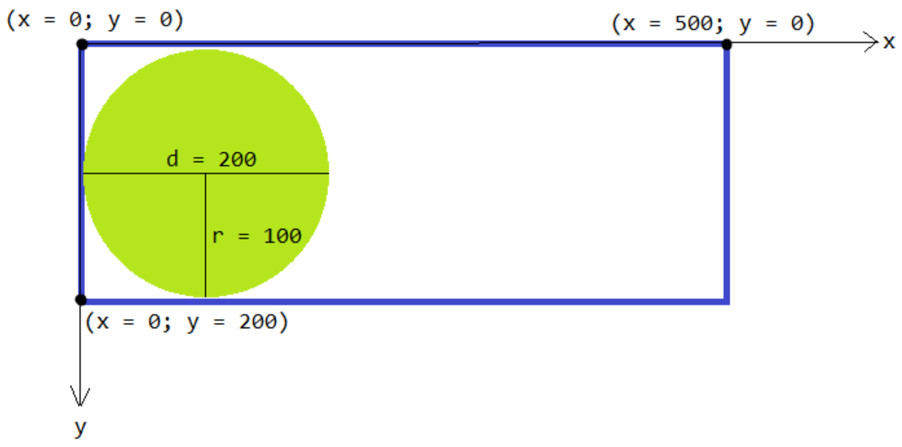
Helpful tips

First of all, please visit the tutorial page https://www.sfml-dev.org/tutorials/2.5/ and the environment setup page https://www.sfml-dev.org/tutorials/2.5/start-vc.php (for Visual Studio).

You can try the example from the lecture, also please change the FPS value to understand how it effects the rendering process.

```
#include <SFML/Graphics.hpp>

int main()
{
    // Create a window
    sf::RenderWindow window(sf::VideoMode(500, 200), "SFML works!");

    // The desired FPS. (The number of updates each second).
    const float FPS = 600.0f;
    window.setFramerateLimit(FPS);

    // Create a circle with radius = 100
    sf::CircleShape shape(100.f);

    // Set the initial color
    shape.setFillColor(sf::Color::Green);
    // Infinite loop, to change
    while (window.isOpen())
    {
        sf::Event event;
        while (window.pollEvent(event))
        {
            if (event.type == sf::Event::Closed)
                window.close();
        }

        // Process Right key button pressed event
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Right))
        {
            // How to change the color
            shape.setFillColor(sf::Color::Yellow);

            // Move to a new position
            shape.move(0.1f, 0.f);

            // Get the current position
            sf::Vector2f current_position = shape.getPosition();

            // Return to the initial position
            if (current_position.x > 300)
            {
                shape.setPosition(0.f, 0.f);
            }
        }

        window.clear();
        window.draw(shape);
        window.display();
    }

    return 0;

}
```

The coordinates system in SFML is below (drawing the above code example):



Control questions

1. How OOP principles are used in your rendering engine? Show some examples.
2. Is SFML library written using OOP approach? How can you prove that?
3. Have you used any design pattern?
4. How did you set up the SFML environment? Which approach did you choose – with dynamic or with static libraries? What is the difference between these two approaches?
5. Why do we need header files and **no** libraries for compilation?

Evaluation

Applying OOP concepts and design patterns	5
Rendering Engine logic	5
Additional points for GUI appearance	Up to 2 points
Total	10+2

Please note that to get the evaluation point will be reduced because of incorrect answers on the questions, wrong program output and logic, in case of Git project absence etc.

Links

C++ Primer 5th Edition: Theoretical part
GoF: Design Patterns
https://www.sfml-dev.org/ : Framework for Game GUI development