# E-Portfolio Instructions Page

## Preamble

In this document I will explain what the two programms in the repository do and also talk about the code in detail. How to setup SFML you can check out the slides, where I have given instructions on how to get started with it.

## Empty.sln

This programm was created to demonstrate how to create a simple window with SFML and render a Circle on it.

### Code Explanation

Just as in every default C++ program we gonna step into a main-routine. The first thing we want to do is to create a Window itself:

```
sf::RenderWindow win(sf::VideoMode(500, 500), "SFML Portfolio");
// It has a size of 500x500, with the title 'SFML Portfolio'
```

After that we have to create our Circle Shape we want to render:

```
    // Creating the object itself
    sf::CircleShape circle;
    // Setting radius to 15pixels
    circle.setRadius(15);
    // Setting FillColor
    circle.setFillColor(sf::Color::Cyan);
    // Changing origin to center
    circle.setOrigin(circle.getRadius(), circle.getRadius());
    // Setting position on windows Center
    circle.setPosition(sf::Vector2f(win.getSize()) / 2.0f);
```

After we have created everyting we need, we can enter our main-loop (In games it's called the game loop) and handle the rendering of objects and the events:

```
    while (win.isOpen()) // Stay in the loop while the window is open
    {
        sf::Event ev; // Create an Event
        while (win.pollEvent(ev)) // Check whether an Event happens on our
window
        {
            if (ev.type == sf::Event::Closed) // Get the Type, if it's the
"Closed" Event close the window
```

```
            {
                win.close(); // Close the window
            }
        }
        ////////////////////////////////// Event handling over

        win.clear(sf::Color::White); // Clear the window every frame with a
white color (Default is sf::Color::Black)

        // <<---
        win.draw(circle); // Draw our Circle object on the window

        win.display(); // Render the window
    }
```

## ml_BouncingBall.sln

This is the solution for the bouncing ball exercise I gave in the presentation. The code is pretty well commented, so I wont explain it any further.