PS0 Hello World with SFML

Setting up a build environment on a virtual Linux system and utilizing it to demonstrate the Simple and Fast Multimedia Libraries was the first job (SFML). We were to alter the SFML demo code to have it create our own sprite and respond to keyboard input when we got Linux up and running.

Key Concepts

Setting up a build environment on a virtual Linux system and utilizing it to demonstrate the Simple and Fast Multimedia Libraries was the first job (SFML). We were to alter the SFML demo code to have it create our own sprite and respond to keyboard input when we got Linux up and running. There was just one main function in this code, and it was not designed in an object-oriented manner. Because the primary goal of this project was to familiarize myself with the Linux development environment and SFML, no notable algorithms or data structures were employed.

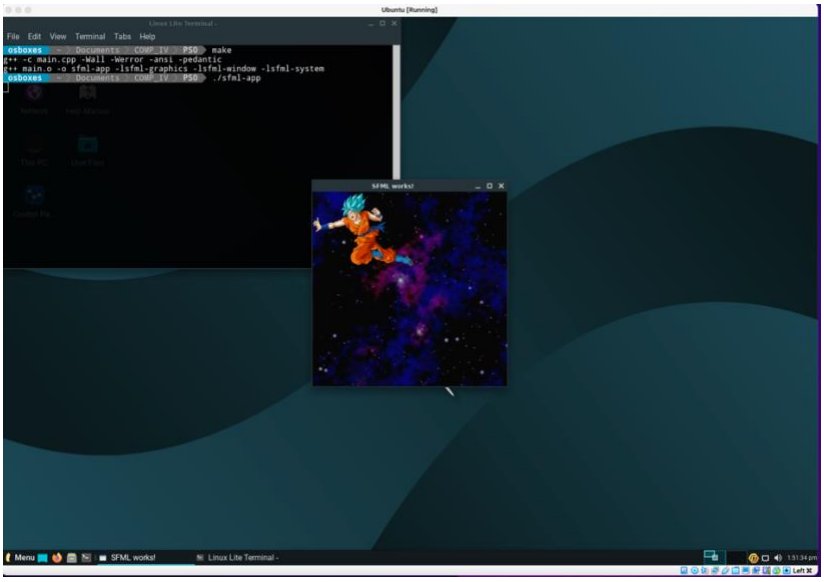
What I accomplished

I started with the SFML instructional website's sample code, which produces a window with a green circle drawn in the center when compiled and run. This project required me to construct a depiction of one of my favorite anime characters in space. I started by making the window 400x400, after which loading in my own sprite images to replace the green circle with a Goku sprite and a space like image. When you hit a directional key, the Goku sprite will travel in that direction.

What I Learned

I learned a great deal about Linux and how to compile programs using Makefiles and the Bash shell, along with setting up a virtual machine. I also learned about SFML and how it can be used to create cross-platform programs with rudimentary graphics, event handling, and audio.

Output



```
    1: CC = g++
    2:
    3: all:
    4:          $(CC) -c main.cpp -Wall -Werror -ansi -pedantic
    5:          $(CC) main.o -o sfml-app -lsfml-graphics -lsfml-window -lsfml-sys
tem
    6:
    7: clean:
    8:          rm *.o sfml-app *~
```

```
 1: #include <SFML/Graphics.hpp>
 2: #include <iostream>
 3:
 4: int main() {
 5:
 6:   // Make window
 7:   sf::RenderWindow window(sf::VideoMode(400, 400), "SFML works!");
 8:
 9:   // Load sprite
10:   sf::Texture texture;
11:   if(!texture.loadFromFile("sprite.png"))
12:     return EXIT_FAILURE;
13:   sf::Sprite sprite(texture);
14:
15:   // Background
16:   sf::Texture star_texture;
17:   if (!star_texture.loadFromFile("starfield.jpg"))
18:     return -1;
19:   sf::Sprite background(star_texture);
20:
21:   while(window.isOpen()) {
22:     sf::Event event;
23:     while(window.pollEvent(event)) {
24:       if(event.type == sf::Event::Closed)
25:         window.close();
26:     }
27:
28:     window.clear();
29:
30:     float offsetX = 0;
31:     float offsetY = 0;
32:
33:     // Get Sprite's current position
34:     sf::Vector2f pos = sprite.getPosition();
35:
36:     // Move image around screen as long as to not move it off)
37:     if(sf::Keyboard::isKeyPressed(sf::Keyboard::Left) && pos.x != 0)
38:       offsetX = -1;
39:     else if(sf::Keyboard::isKeyPressed(sf::Keyboard::Right) && pos.x != 4
00 - 198)
40:       offsetX = 1;
41:     else if(sf::Keyboard::isKeyPressed(sf::Keyboard::Up) && pos.y != 0)
42:       offsetY = -1;
43:     else if(sf::Keyboard::isKeyPressed(sf::Keyboard::Down) && pos.y != 40
0 - 152)
44:       offsetY = 1;
45:     else if(sf::Keyboard::isKeyPressed(sf::Keyboard::R)) {
46:       sprite.setPosition(0, 0);
47:       pos.x = pos.y = 0;
48:     }
49:     else if(sf::Keyboard::isKeyPressed(sf::Keyboard::Escape))
50:       window.close();
51:
52:     // Set a new position
53:     sprite.setPosition(pos.x + offsetX, pos.y + offsetY);
54:
55:     // Draw images
56:     window.draw(background);
57:     window.draw(sprite);
58:
59:     window.display();
60:   }
61:
62:   return 0;
63: }
```