Department of Computer Science and Engineering
University of Dhaka
31 December, 2022

Project Report:
Fundamentals of Programming Lab(CSE-1211)

Project Name:
SAVING THE WILD

Team Members:
Meherun Farzana (Roll: 05)
Tasnia Iffat (Roll: 27)

1. Introduction

"Saving the Wild" is an SDL-based adventure storyline game written in C++ programming language. There are three distinct levels in this game- a driving level and a running level. In the first two levels, the player explores a picturesque place, collects points by cleaning up the magnificently beautiful place and avoiding obstacles, and defeating enemies. In the third level, the player collects gems and rewards for being a savior of mother nature. It is a simple but highly enjoyable game with a crucial message about the conservation of nature.

2. Story

Nature has been an integral part of human life since the beginning of time, providing us with the basic necessities of survival such as clean air, water, and food.On an environmental level, forests play a vital role in the health of the planet. They act as carbon sinks, storing carbon dioxide and helping to mitigate the impacts of climate change. They also provide habitat for a wide range of plants and animals, supporting biodiversity and the health of ecosystems. Forests also have cultural and recreational value, providing opportunities for recreation and tourism, as well as serving as important cultural and spiritual sites for many communities. Overall, forests are an invaluable resource that should be protected and preserved for the benefit of both the environment and humanity.

However, nowadays every natural habitat is suffering the consequences of human being's ignorance and negligence. The land, water, air, and even picturesque forests are contaminated with organic and inorganic waste. Inorganic waste especially brings bad impacts on the forest ecosystem. As it is difficult to decompose, the accumulation of inorganic waste potentially makes the soil layer impenetrable by plant roots. Our game is a fun way of increasing awareness about the importance of the conservation of nature.

With this goal, we have created three levels in our game. The first level is a running level, where the player explores the forest directly and collects inorganic waste. The player also has to defeat the enemy hunters who try to attack the wildlife of the forest.
The second level is a driving level where the player has to drive a car through a beautiful forest road. The player earns points by picking up inorganic waste(cans,batteries, plastic and polythene waste) and avoiding cars and vehicles approaching from the opposite direction.
In the third level, the player reaps the rewards of being kind towards nature by collecting treasures like valuable gems, quartz, crystals, etc.

3. Objectives

1. Making a simple and easy, yet fun and interesting adventure game.
2. Attractive graphics and animations along with interactive sound.
3. Well-organized modular code suitable for further development.
4. Paying tribute to mother nature and increasing awareness about conscious use of natural resources.

4. Project Features

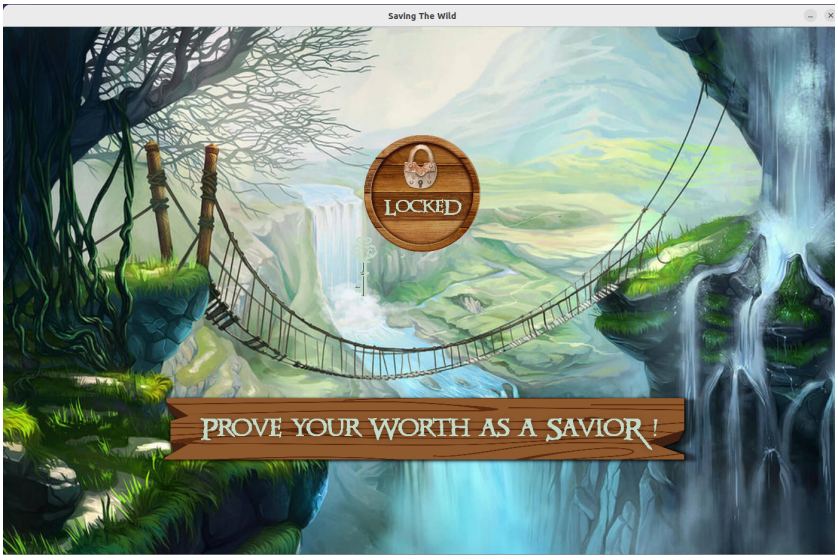
1. Simple and easily customizable modular code structure with proper commenting.

include > C headers.h

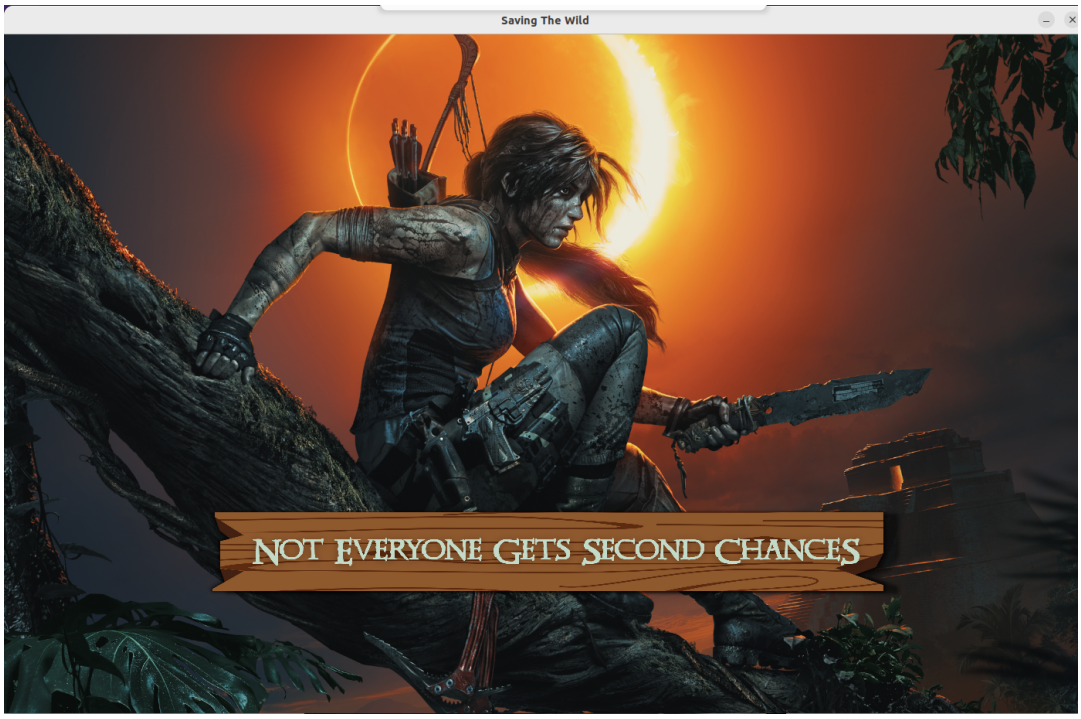
```c
#pragma once

#include "preprocessors.h"
#include "constants.h"
#include "car.h"
#include "collider_cave.h"
#include "collider_level_one.h"
#include "collider_level_two.h"
#include "collision_checker.h"
#include "init.h"
#include "loadstages.h"
#include "mouse.h"
#include "playerinfo.h"
#include "shooting.h"
#include "shutdown.h"
#include "sprite.h"
#include "texture.h"
```

2. 3 of the most important features along with other interesting features of our game are:
    a. Level two is locked unless a player succeeds to achieve a sufficient score (for now it is 1200).



b. One player cannot play level one more than twice at a straight to save his/her previous scores.



c. If the player survives for a certain amount of time, the difficulty of each level increases automatically as the forest hunters become more savage.

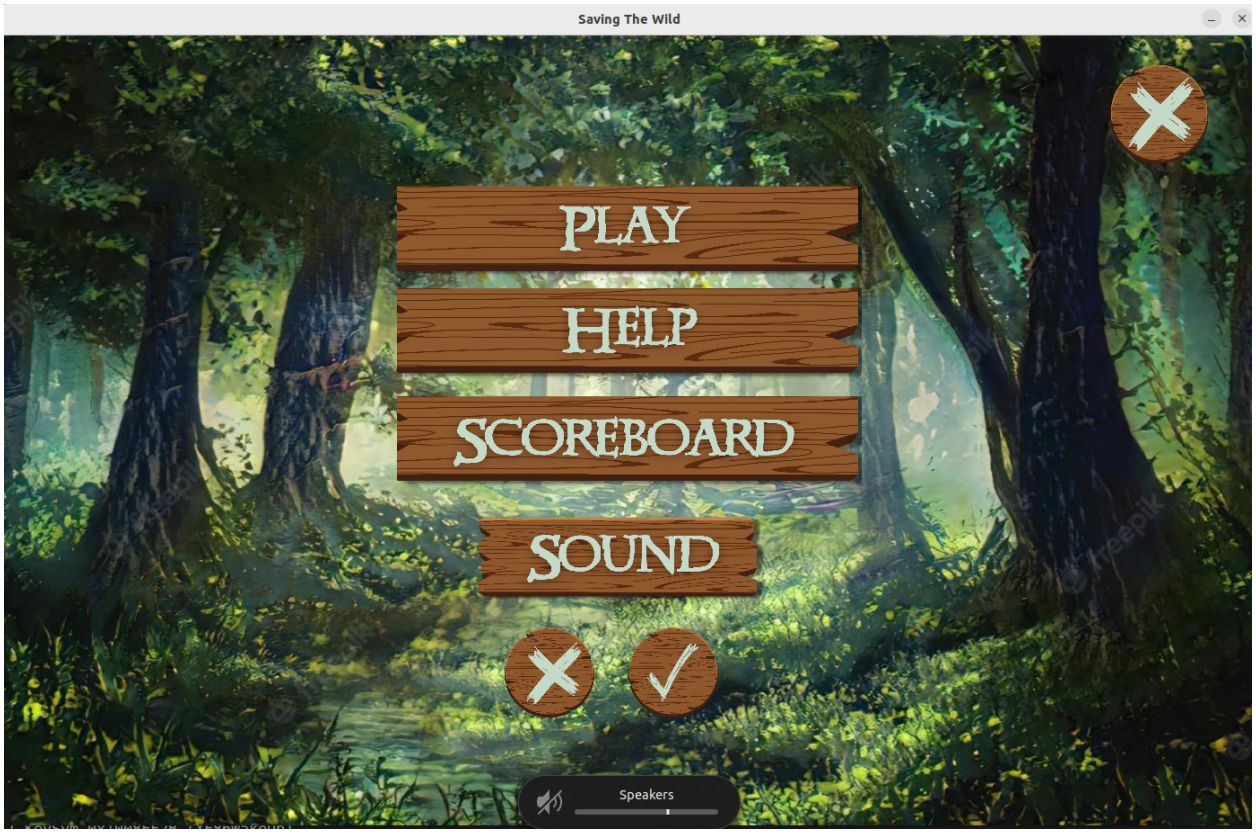
These features challenge the player to cope with the game gradually.

3. Use of “class” and “structure” to make the code more readable and easily understandable to other developers. Besides “class” let us use constructors and destructors along with helping us with public and private variables so that the game doesn't face unwanted changes.
4. Attractive and feature-rich Graphical User Interface. The player has multiple options to interact with the application. This is especially convenient for users who are not comfortable with the command line interface.

5. Easy build and run command in the makefile to start the game using one command only “make game”. We have also included separate debug and release commands “make debug” and “make release” for players who are more comfortable using the CLI and want to customize the game to their liking.

6. Beautiful customized game icons and screens.

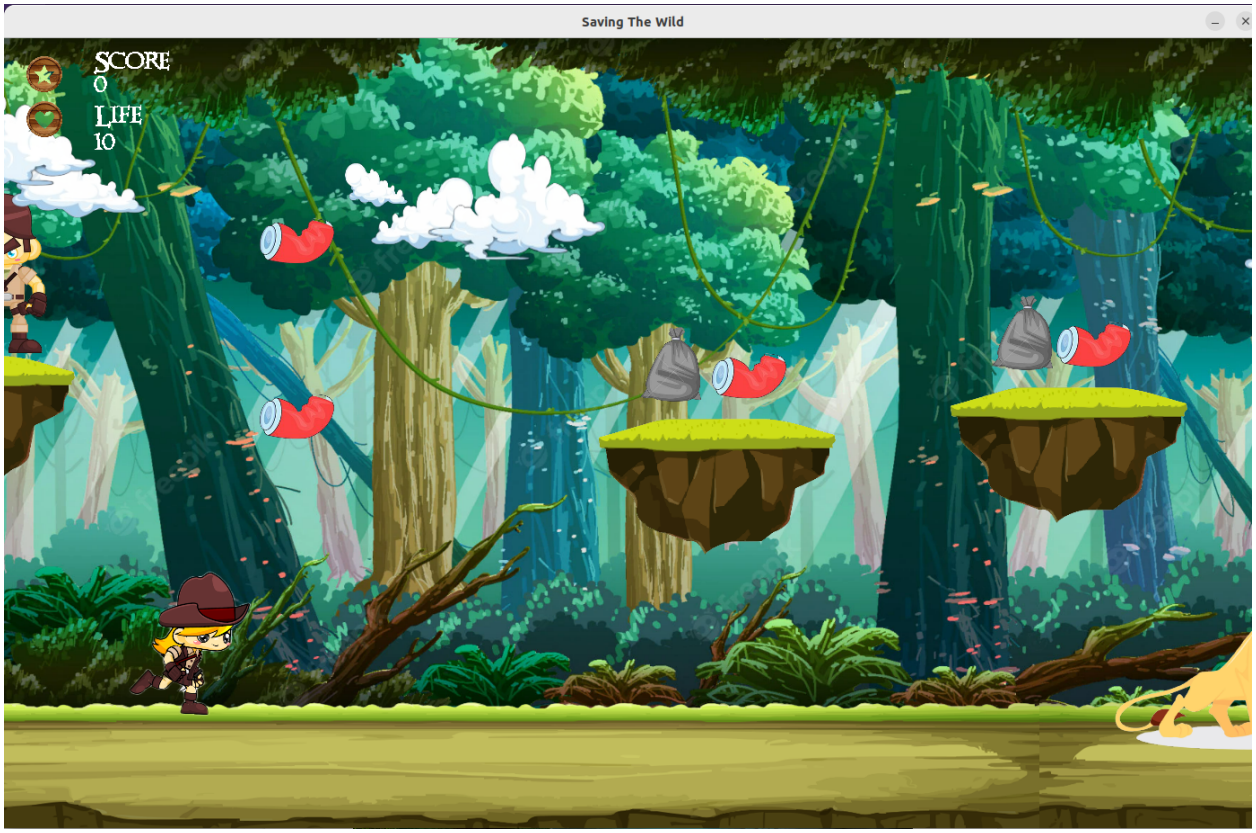
7. Attractive and interactive menu to control the game.



8. Help page to guide the player and provide instructions.

9. Proper use of files makes the game competitive among peers.
10. Increasing difficulty and complexity in various levels. Unique events for various objects on the screen. The user may gain points, gain lives, or lose lives depending on the collided object.
11. Adventurous and soothing music to enhance the gameplay experience. Easy sound and music on/off feature.

12. Clear and dynamic score and life feature on the display in each level.



13. GitHub repository with helpful resources.

5. Project Modules

Our code base relies on the following custom header libraries:

- preprocessors.h
  In this header file, we included the standard c++ library and relevant SDL libraries. By using this header, we avoided manually including the libraries in each file.
- headers.h
  Here, we have included all of the custom header libraries of our project so that we could include all of them easily in the source cpp files.
- constants.h
  In this file, we have declared extern variables used in other modules of the project for easy access.Some examples of our global variables are screen dimensions, score and life variables, buttons for screen navigation, music and textures.
- init.h
  Here, we have declared the functions to start up the game. init() function initializes the game and the main flow of the game occurs inside gameloop() function.

- texture.h
  We have declared a custom struct for all the textures throughout the game. The struct contains information about texture dimensions, functions to load images and fonts from file path, render them on the screen etc. We have also declared some extern textures here which were used in other modules.
- sprite.h
  We have used sprite sheets for running, jumping, and shooting motion of our player. For this, we created a custom class in this module which includes their positions, velocity, movement and event functions.
- shooting.h
  Here, we declared a function to handle shooting by the player in the forest and cave level.
- car.h
  This header contains a custom struct for the player car in the driving level. Using this struct, we handled movement and collisions of the car.
- collider_level_one.h
  Here, we have a custom struct to render all the SDL_Rects which can undergo collision in the forest level.
- collider_level_two.h
  Here, we have a custom struct to render all the SDL_Rects which can undergo collision in the driving level.
- collider_cave.h
  Here, we have a custom struct to render all the SDL_Rects which can undergo collision in the cave level.
- collision_checker.h
  In this header library, we have declared a function to check if any collision has taken place in all the levels. Based on collision, the player earns points, collects and loses lives. Game Over also depends on this function.
- mouse.h
  Here, we have created a custom struct for all the buttons which were accessible by mouse. This struct includes the position of the buttons, function to detect mouse click and handle consequent events.
- playerinfo.h
  Here, we have created a custom class to handle File i/o. Using this class, we stored name of the player and their score. And we displayed the highest scorers in the scoreboard screen.
- stages.h
  Here, we have declared all the separate screens to be displayed throughout the game using an enum. In this way, we made screen navigation smooth and easy. It is also easy to add a new level or new screens for further development.
- loadstages.h

This is one of our most integral header libraries. Here, we have declared functions which render textures, fonts, music for each unique stage declared in the previously mentioned enum.

- shutdown.h
  Here, we have declared a void function to free all music, images, objects and text to avoid memory leakage. We also quit the SDL libraries here before closing the game.

6. Team Member Responsibilities

Tasnia Iffat

Roll: RH-27
Registration Number: 2020-415-637
GitHub LinkedIn

1. Ideation, Logic Design and Implementation of Game Loop
2. Structured Game Source Code writing in C/C++ (Level two and three)
3. Version Control (Git/GitHub)
4. Graphics Designing (Photoshop/Canva)
5. Modules: preprocessor.h, headers.h, constants.h, init.h, car.h, collider_cave.h, collider_level_two.h, stages.h, texture.h, loadstages.h, shutdown.h ,sprite.h

Meherun Farzana

Roll: RH-05
Registration Number: 2020815615
GitHub

1. Structured Game Source Code writing in C/C++ (Level one)
2. Logic Design and Implementation of Game Loop
3. Code Testing and Bug Fixing
4. File Handling
5. Modules: headers.h, constants.h, collider_level_one.h, check_collision.h, texture.h, sprite.h, shooting.h, playerinfo.h, mouse.h (Level One)

7. Platform, Library & Tools

1. C/C++ - The code of the project is written in this powerful general-purpose language.
2. SDL2 - Simple DirectMedia Layer is a cross-platform development library designed to provide low-level access to audio, keyboard, mouse, joystick, and graphics hardware.
3. Visual Studio Code - A powerful, free, open-source IDE that runs on all platforms.
4. Git/GitHub - Git is an open-source version-control tool. GitHub is a code hosting platform for easy collaboration.
5. Photoshop -Photoshop is an image creation, graphic design, and photo editing software developed by Adobe. The software provides many image editing features for pixel-based images and vector graphics.

6. Canva- Canva is a free-to-use graphic design tool.

8. Limitations

Due to our limited knowledge of graphics designing and limited time, we could not create custom graphics for every single screen. Our icons, buttons, driving and cave level backgrounds are custom made but in other screens, we had to rely on stock images. We also intended to implement a locked/unlocked feature for level two and level three but we failed to implement this feature. We intend to debug and update this feature later on.

9. Conclusions

By working on this project, we learned more about C/C++ and SDL. But more importantly, we learned collaboration, peer communication, organized and structured programming and pressure handling. Several times throughout the project, we got stuck and navigated various websites to find a solution and fix those bugs. Given the internet resources mostly use modern APIs,since SDL is a very old library, we had to dig deeper to find what we actually needed and in some cases we had to figure it out ourselves through hours of trial and error. We also had to figure out how to tie all the modules together in a coherent manner. We feel more confident about learning new technologies and solving problems on our own. We believe all of these will be beneficial for our future in the field of software engineering.

10. Future plan

We want to create fully custom graphics and music to make the project entirely our own. We also have aspirations to build this game across various platforms (Android, iOS, Windows) in the future. We believe it would be a great opportunity to learn to work on different platforms.

Repositories

GitHub Repository: https://github.com/tasniaiffat/CSEDU-Game-Project

Youtube Video:https://www.youtube.com/watch?v=o07BmouNg2w

References

1. https://lazyfoo.net/tutorials/SDL/index.php
2. https://wiki.libsdl.org/Tutorials
3. http://www.sdltutorials.com/
4. SDL Basics

5. Free Graphics Resources
6. Modular Programming basics
7. Git and GitHub tutorial
8. Graphics Coordinate navigation
9. Level 2 Music
10. Level 1 Music
11. Sound chunks:
    - https://www.youtube.com/watch?v=i0DON3AjhW4&ab_channel=SoundEffects
    - https://www.youtube.com/watch?v=B14L61fYZlc&ab_channel=Lupi
    - https://youtu.be/mIfw1a7E56g
    - https://www.youtube.com/watch?v=HCqRNkiE0lI&ab_channel=CCHub-FreeMusicandFootages
12. Game tutorial: Beginner friendly resource to create a modular SDL Project.
13. Stack Overflow - Where Developers Learn, Share, & Build Careers : helpful for understanding errors and debugging code.