My GitHub Project
C++ & SDL2

ConsoleArt

November 19, 2025 **Riyufuchi**

Contents

Glossary

SDL2 A cross-platform low-level multimedia library in C for handling graphics, input, audio, and window management. 1, 4

Acronyms

GUI Graphical User Interface.

1 Introduction

This application originally began as a purely CLI-based project (which is also the reason for its name, *ConsoleArt*). As the project evolved, a Graphical User Interface (GUI) was later added. Development started at a time when C++17 was the newest standard, so the use of modern features such as smart pointers was adopted gradually throughout the project.

1.1 Technologies Used

1.1.1 SDL2

This project is primarily written in C++ and uses SDL2 for window management, rendering, and input handling. Since SDL2 does not provide native GUI widgets, the application uses a custom library to implement UI components on top of SDL.

1.1.2 STB

The project utilizes the single-header `stb` libraries, primarily `stb_image`, to load image formats such as PNG, JPEG, and HDR files.

1.1.3 TinyFileDialogs

`TinyFileDialogs` is used to provide simple, cross-platform file selection dialogs, allowing users to select images without relying on large external GUI toolkits.

1.1.4 ConsoleLib

`ConsoleLib` is my custom library developed to simplify the creation of CLI applications. It provides an abstract `IConsole` interface and several platform-specific implementations:

- **DefaultConsole** – A basic implementation using standard output (`std::cout`) without any additional formatting features.
- **UnixConsole** – Adds support for text coloring and formatting through ANSI escape codes commonly available on Unix-like systems.
- **WindowsConsole** – Inherits from `UnixConsole` and enables UTF-8 output on Windows while ensuring compatibility with ANSI escape codes.

The library also includes additional utility modules such as an argument parser and other helpers commonly required in CLI tools, making it reusable across multiple projects.

1.1.5 ComponentsSDL

`ComponentsSDL` is my lightweight UI component framework built on top of SDL2. Since SDL2 does not provide native GUI widgets, this library introduces a structured way to create interface elements within an SDL application.

The framework provides reusable components such as panels, buttons, and layout helpers, allowing the application to assemble its interface in a modular and maintainable manner. Each component is responsible for its own rendering and event handling, making the system extensible and easy to integrate into SDL-based projects.

2 Core

2.1 Controller

The Controller class is core of this application.

2.1.1 Pointer ownership

The controller owns all loaded images and stores them in vector and thanks to use of unique pointer they do not need to be deleted manually.

```
1  using IndexDataType = uint32_t;
2
3  struct VectorNode
4  {
5    IndexDataType index;
6    std::unique_ptr<Images::Image> imageUptr;
7  };
8
9  using VectorData = VectorNode;
10
11 std::vector<VectorData> images;
```

Listing 2.1: Image vector in Controller class

2.1.2 UML

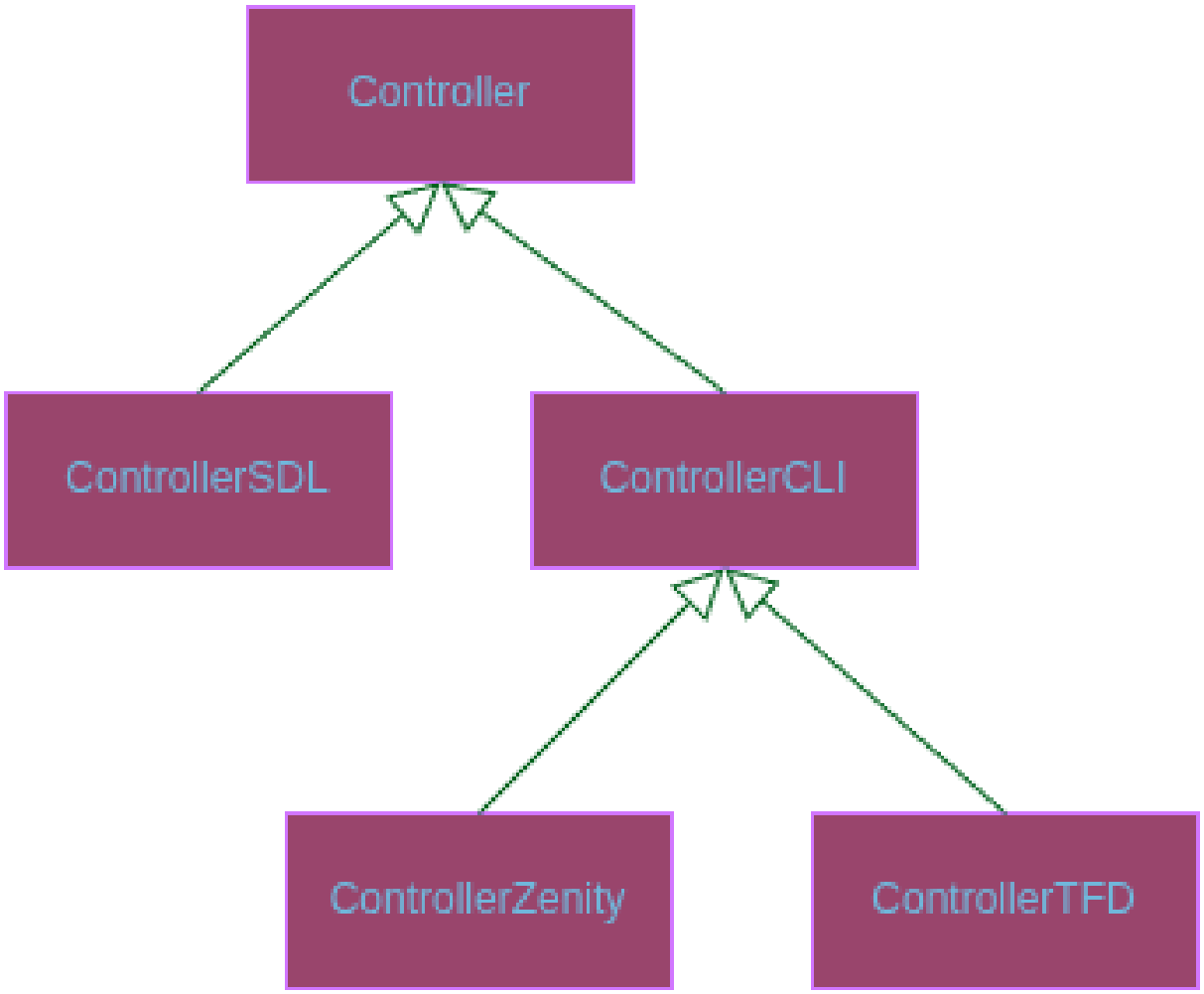

Figure 2.1: Controller inheritance UML

2.2 Controllers childs

2.2.1 ControllerCLI

2.2.1.1 ControllerZenity

2.2.1.2 ControllerTFD

2.2.2 ControllerSDL

3 GUI

3.1 Foreword

3.2 Window

3.3 States

4 Image formats

4.1 Image class

4.2 Interfaces

4.2.1 IMultiPage

4.2.2 IAnimated

Summary

Listings

List of Figures