Task 08 - Spike Summary Report

Spike: Task_08
Title: Game State Management
Author: Thomas Horsley, 103071494

Goals & Deliverables

Aim: To implement Zorkish Phase I, as described in the specification document.

Deliverables:

- Design Documentation
- Functional Zorkish Phase I Solution
- Spike Summary Report
- Git Commit History

Technology, Tools and Resources

Tech and Tools

The project was scripted in C++ 20 using Visual Studio Community 2022.

UML's and charts are made with *www.Lucidchart.com*

Source control is handled using Git.

Resources

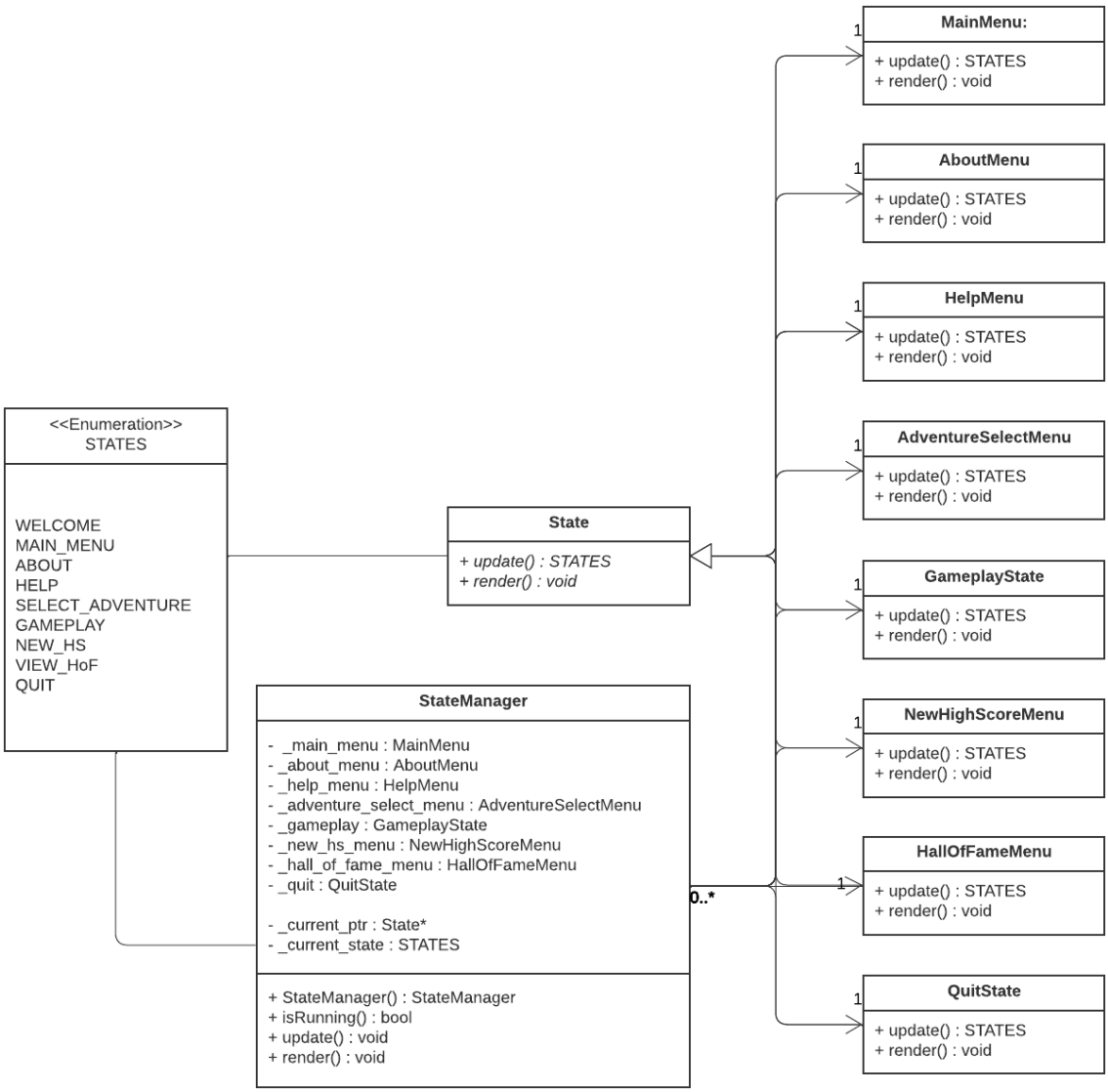
- Lecture *"Game States and Stages"*
- Lecture *"Design Patterns Introduction"*

Task 08 - Spike Summary Report

Tasks Undertaken

Planning

Diagrams and Charts



Task 08 - Spike Summary Report

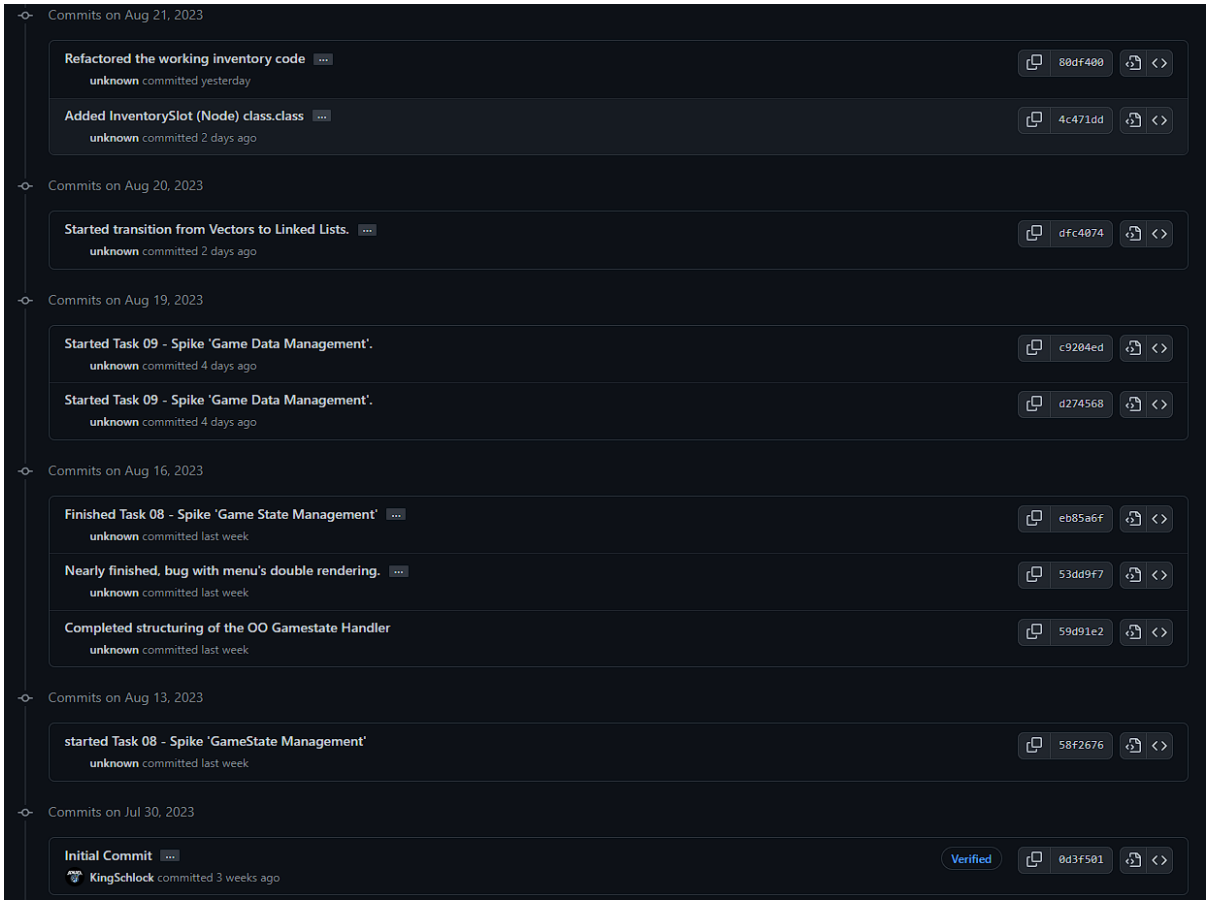
Class Descriptions and Notes

The Zorkish StateManager class was implemented using the state management pattern. An enum of states is contained within the StateManager header, each of which correlates with a State Object child class containing the relevant state functionality.

The StateManager will construct a single instance of each State child and (depending on the value of _current_state) will wrap the appropriate state functionality within it's own *update( )* and *render( )* methods before presenting this to the *main( )* method.

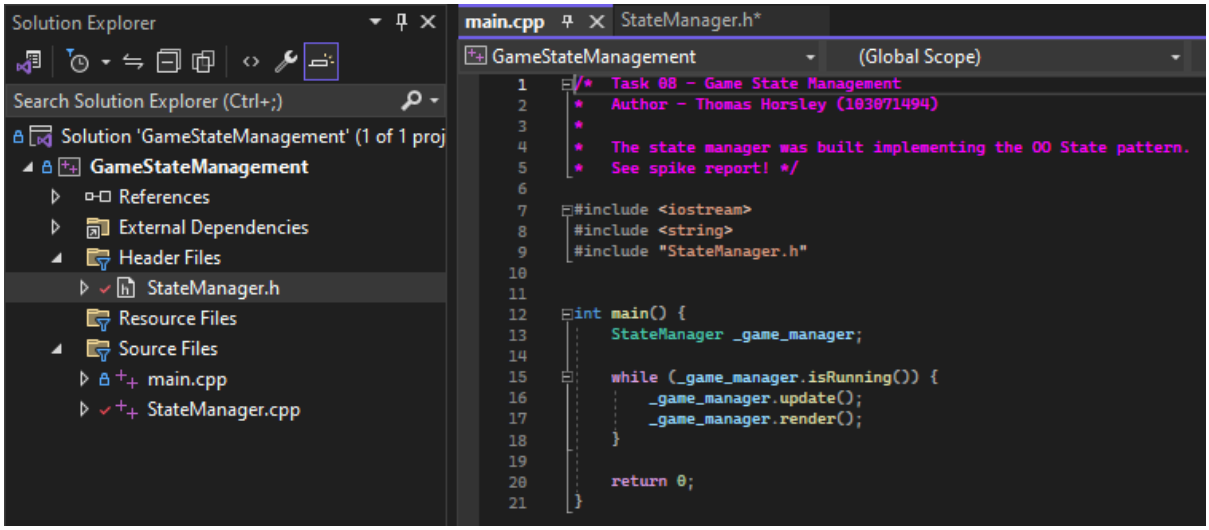
Implementation

Git Commit History

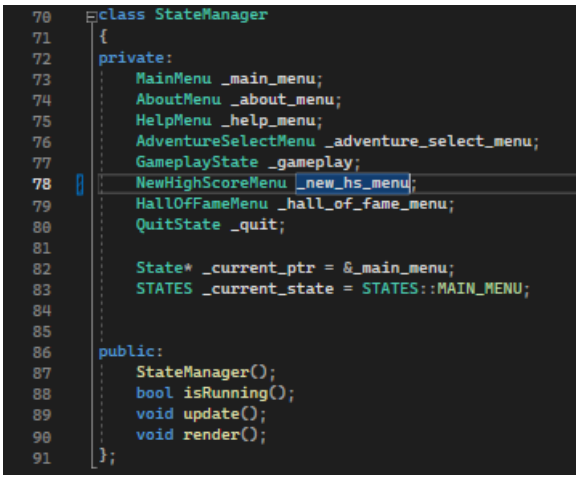


Task 08 - Spike Summary Report

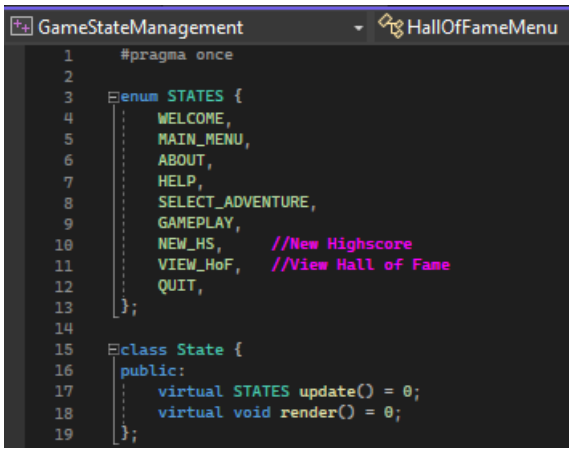
Code Snippets



main( )with an instantiated StateManager object



StateManager definitions



STATES enum and State class definitions

Task 08 - Spike Summary Report

What was Learned?

This task involved creating a scalable state management system for the Zorkish Phase I prototype. The Spike involved manipulating custom classes and return types to effectively manage a state driven game loop. Additionally, a state driven design allows the developer to encapsulate specific state functionality within a manager object, further abstracting the management system from the user.