Aether of Enclaves

Explore the sky.

Samuel Eubanks
McKenzie Weller
v0.0

February 16, 2018

Contents

1 Project Requirements

1.1 Requirements

1. Version Control: GitHub, Travis CI
2. Testing Framework: Stainless
3. Design Patterns:
   a) Singleton (Logger)
   b) Flyweight (Graphics classes)
   c) Command (Input handler)
   d) Prototype (Enemy spawning - TBD)
   e) Observer (Achievements - TBD)
4. User Interface Method: GUI

1.2 Questions

1. **What is your project?**
   Aether of Enclaves will be a 32-bit exploration game, in which the user controls a main character and an airship and travels through the sky - picking up crew members, discovering new islands, interacting with NPCs, and exploring.

2. **What technologies will you use?**
   We intend to write the game in Rust and use a corresponding game engine framework called Piston. The testing framework we will use will be Stainless, which has been developed for Rust. Most, if not all, of the graphics will be produced using Aesprite, a pixel art editor.

   As of current, these are the technologies we anticipate using. We are unfamiliar with Rust as a whole, including its frameworks, and will be learning as we go. We both have a mild exposure to using Aesprite, but this program is rather straightforward.

3. **What are the essential parts of the project?**
   Clearly, since we are creating a game, there must be gameplay for the project to be successful. The essentials for the type of game we are trying to produce specifically are as follows: a main character, saved game data, a variety of NPCs and NPC data, a generated environment, and graphics. As a baseline, we will need to implement each of these even in their simplest forms (e.g., in a console with ASCII art). Of course, we would plan to maintain a large library of NPCs and procedurally generate the world.

4. **What outside resources will you require?**
   There is a possibility of the need for outside artwork (if we are unable to complete enough original art) and sound (background music, etc.). As of right now, this is all TBD.

5. **Make a proposal for your architecture.**
   Our back-end will handle the AI, generation of the game world, data storage (this will be saved locally), game mechanics, user input, logging, and interaction within the game world (e.g. with items or NPCs). This will not be written using a specific framework.

   Our front-end will be the user interface and game graphics respectively. This will be created using the Piston game engine, which allows incorporation / production of graphics.

   The front-end and back-end are in many ways incorporated in the application (i.e. we do not have something like a server / database that would require an RPC). They will not be asynchronous as a result. The back-end will load saved data locally (as mentioned previously) which is savestate functionality included with the game engine.

   The design patterns we will have are listed in 1.1.1. Objects will use composition over inheritance. Other objects include: Player, Creature (for NPCs / Enemies), Airship, World, and Game. (I would predict that we'll develop a few more as we move forward). The interface will be developed in a separate file. There will additionally be structs for Items and smaller objects.

6. **Detailed Plan**
   2/28
   - ☐ Generate basic graphical interface with simple menu, using Piston (one class/file).
   - ☐ Beginnings of the Command class (input handler) written and incorporate with above.
   - ☐ Have skeleton of Game loop written.
   - ☐ Have a list of NPCs/items we would like to include.

   3/21
   - ☐ Implementation of Airship and Player class integrated with the Game.
   - ☐ Graphics code for Airship and Player written in Flyweight class.
   - ☐ Testing for Airship, Player, and Input class developed.
   - ☐ If possible, beginnings of enumerations for NPC classes.

   4/4 (after Spring Break)
   - ☐ Successful inclusion of savestate, if not yet completed.
   - ☐ Testing and completion of the world generation we will be submitting.
   - ☐ When available, further development on Game class.

   4/18 -
   - ☐ Continuing to add items, NPCs, graphics, and other features.
   - ☐ User testing, and then some more user testing.
   - ☐ Tie up any loose ends for final submission.

   5/2 - Submission of game.

2 Overview

2.1 Main Concept

Aether of Enclaves will be a 32-bit exploration game, in which the user controls a main character and an airship and travels through the sky - picking up crew members, discovering new islands, interacting with NPCs, and exploring.

2.2 World/Environment Theme

The environment of AoE is a steampunk-themed alternative world, with industrial and magical influences. Its distinguishing characteristics include floating islands / landmasses, inhabited by non-humanoid creatures.

2.3 Objective of the Game

The objective is to explore, expand your crew and ship capabilities, and collect resources to do so. It is very much a "make your own adventure" style game.

2.4 Art Style

Art and graphics will primarily be 32-bit pixel art. In the future, we may incorporate conceptual art. Artwork will integrate a specific color scheme, utilizing earthy tones and low-saturated colors.

2.5 Controls

The user will have controls to: move, open a menu, direct the ship, and interact with the environment (in the form of a general 'action' button). TODO: Include a diagram / prototype of the control interface.

3 Mechanics

Things to have

- Have basic items
- Have NPCs
- Be able to travel to different islands
- Have ship (base)
- Be able to recruit people

Not all of the following are planned to be developed in the scope of CSCI 3010.

3.1 Base

- An airship which is upgradeable and "levels up", rather than the main character
- Conveyor belts / mine tracks can be installed around base to increase productivity, for a cost
- Airship is made of modular parts that can be replaced / upgraded
- Different components together create synergies
- Ship has a "heart" which needs to be defended

3.2 Crafting

- Gathered resources can be used in production of upgrade parts, potions, food, and other items
- Crafting through use of crewmates increases their range of abilities

3.3 Creatures

- NPCs are non-humanoid creatures, which can either be friend or foe
- No direct words / conversations are exchanged between MC and creatures - instead pop-up "balloons" (icons) are used to indicate expressions and communication

3.3.1 Main Character

- By default does not have many skills
- Actions include: Recruiting crew, coordinating upgrades and battles, commanding the crew, collecting items

3.3.2 Crew

- NPCs in the world can be recruited to your team after a certain requirement is met
- Play-style is determined more by crew than main character (i.e. the type of crew determines the play-style instead of the MC having certain abilities - such as a fighter, mage, cook, and so on)
- Actions include: Crafting, developing skills

3.3.3 Monsters

- Monsters are found around the world (both passive and aggressive)
- Monsters can be captured / used on the airship (e.g. for battles), as well as be trained to be stronger
- Different monsters can perform different tasks (produce crafting ingredients, work, fight, etc.)

3.3.4 Robots

- Robots and other machinery can be created to automate resource gathering, crafting, transportation, and fighting

3.4 Combat

3.4.1 Ship Combat

- Cannon-based fighting using items as ammo
- TODO: To be developed further

3.4.2 Player Combat

- Top-down, real time combat
- TBD: Separate combat screens

3.5 World

3.5.1 Floating Islands

- Some islands are more exploration focused, some are farming focused, some are combat focused, etc.
- Islands may or may not be inhabited with creature settlements

3.5.2 Caves

TODO: Currently of scope for CSCI 3010

3.5.3 Dungeon

TODO: Currently out of scope for CSCI 3010

3.5.4 Settlement

TODO: Current out of scope for CSCI 3010

3.6 Agriculture

- Islands may posses a certain soil quality which allows varying plants to thrive
- Seeds can be obtained by eating food or converting plants into seeds
- TODO: Further development on Agriculture

3.7 Items

- Items may be acquired via crafting, interaction with NPCs (fighting, trading), or exploration
- Items may be disposable (food, potions, seeds), or non-disposable and have potential to be leveled up (ship components, weapons, other tools)

4 Technology

4.1 Target Systems

Windows / Mac / Linux

4.2 Hardware

Monitor, Keyboard / Controller

4.3 Development Systems/Tools

Language: Rust
Libraries: Piston, Stainless
Graphics tools: Aesprite, Gimp