Version: 1.9

FACADE

2D Combat-Action Game Project

By:
Omid Aran - Leslie Hoyt - Bezan Lilauwala - Faizan Mohammed

Facade
Architecture/Design Document

Table of Contents

Change History

Version: 1.9
Modifier: Bezan Lilauwala
Date: 5/17/2021
Description of Change: Added player ability to defend and added new type of enemy, along with changes to Level 3 (added Projectile.cs, ProjectileEnemy.cs)

Version: 1.8
Modifier: Omid Aran, Faizan Mohammed, and Leslie Hoyt
Date: 5/16/2021
Description of Change: Bug fixes

Version: 1.7
Modifier: Bezan Lilauwala, Omid Aran, and Leslie Hoyt
Date: 5/15/2021
Description of Change: Bug fixes

Version: 1.6
Modifier: Leslie Hoyt
Date: 5/15/2021
Description of Change: Added Boss1 Level (added Boss1.cs, Boss1LevelMaster.cs, SpellAttack.cs and SpellFX animation) and fixed bugs

---

Version: 1.5
Modifier: Bezan Lilauwala
Date: 5/13/2021
Description of Change: Added Level 3 (added MeleeEnemy.cs)

Version: 1.4
Modifier: Leslie Hoyt
Date: 5/02/21
Description of Change: Added pause functionality (added Pause.cs and changes to PlayerMovement.cs, PlayerCombat.cs, EnemySamurai.cs, EnemyWizard.cs, EnemyWorm.cs)

Version: 1.3
Modifier: Faizan Mohammed
Date: 5/13/21
Description of Change: Added Level 1 and Level 2

Version: 1.2
Modifier: Faizan Mohammed
Date: 4/13/2021
Description of Change: Provided system behavior, logical view and user-case details

---

Version: 1.1
Modifier: Faizan Mohammed
Date: 4/10/2021
Description of Change: Initial rough draft. Addition of introduction & design goals

1 Introduction

This document describes the architecture and design for the 2D game application; Facade, catered to players who would want to play games online. The application is a 2D combat & platformer game where the player can beat as many enemies as they can for a high score until they reach the end of the level. Users are able to play up to 3 levels as they complete each level in succession. The game's medium can be compared to old flash games that can be accessed online from around 2008.

The purpose of this document is to describe the architecture and design of the Facade application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

- Users/Customer – they want assurances that the architecture will provide for system functionality and provide the gameplay satisfaction they are yearning for.
- Developers – they want an architecture that will minimize complexity and development effort. Specifically to understand the code and boost reusability.
- Project Manager – the project manager is heavily responsible for meeting the requirements of the application of this project. Their insight and judgement is needed upon the game's pacing and enjoyability. Constructive feedback and criticism from project managers alongside with the programming team will lead to a fleshed out and well-developed application.
- Maintenance Programmers – they want assurance that the system will be easy to evolve and maintain on into the future. The program's main architecture should be easy to adapt and maintain for updates and additions of the game.

The architecture and design for this game project can be intricate for stakeholders and developers who have a different vision in mind for the game. The game's architecture is a basic 2D combat-platformer with the potential to have many features added. Presenting the game's architecture in different perspectives can pave the way for stakeholders and developers to pursue the endeavor to envision their ideas for the game. The application can be split in several views:

1. Logical View – major components, their attributes and operations. This view also includes relationships between components and their interactions. When doing OO design, class diagrams and sequence diagrams are often used to express the logical view.
2. Process View – the threads of control and processes used to execute the operations identified in the logical view.
3. Development View – how system modules map to development organization.

4. Use Case View – the use case view is used to both motivate and validate design activity. At the start of design the requirements define the functional objectives for the design. Use cases are also used to validate suggested designs. It should be possible to walk through a use case scenario and follow the interaction between high-level components. The components should have all the necessary behavior to conceptually execute a use case.

2 Design Goals

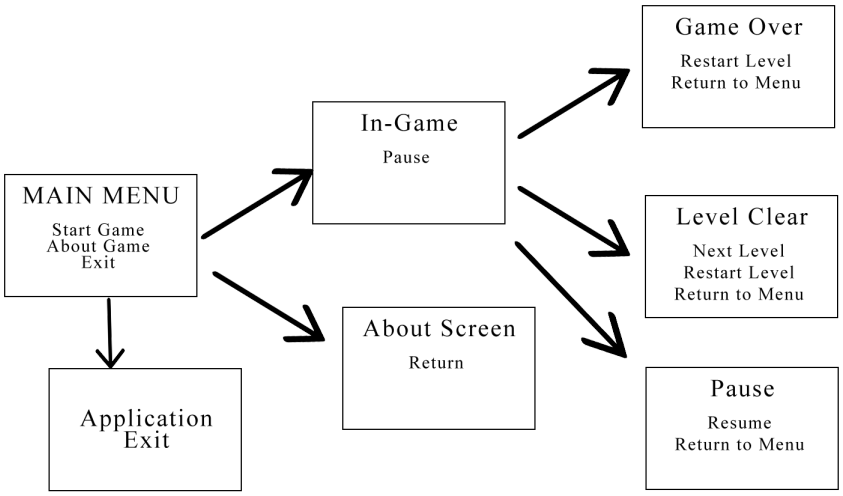
The design goals held in regard for this project were held fairly lightly. Due to time constraint, we have only followed through with completing designs that would make the foundation of the game. Stakeholders and developers can build off the laid groundwork of the application and turn it into the product that they want. Development that followed through in the application turned out to be a great learning experience for the team, as we delved into the 2D environment of the Unity engine. The game's design practices are simple and clear for what we have envisioned it to be. We went for a 2D combat-platformer that focuses on the player to achieve a higher score with the completion of each level by defeating as many enemies as they can.

The design priorities for the Facade application are:

- The design should be clean to follow through to boost development.
- The design should be able to have support for addons for future development. Given Unity's 2D architecture and IDE, developers are able to bring in a multitude of functionalities from 3rd party plugins or from the development itself. Objects and scripts should be able to provide basic support for the application, where developers in the future can branch off what's established and adapt them further.
- The design should provide some sort of reusability, where a basic script's functionalities can be applied to others to reduce redundant code.

3 System Behavior

The game's system behavior follows a basic video game startup format where the player can proceed to play along with other basic functionalities most games have since the beginning. The following diagram outlines the system behavior the game follows, more details can be disclosed in other subsections below:



4 Logical View

The logical view describes the main functional components of the system. This includes modules, the static relationships between modules, and their dynamic patterns of interaction.
The logical view will entail the game's operation and functionality. It will primarily focus on two aspects; a high-level and mid-level. Since the complexion of this project is still at its base, there are still vital functions that need to be highlighted to understand the application's process.

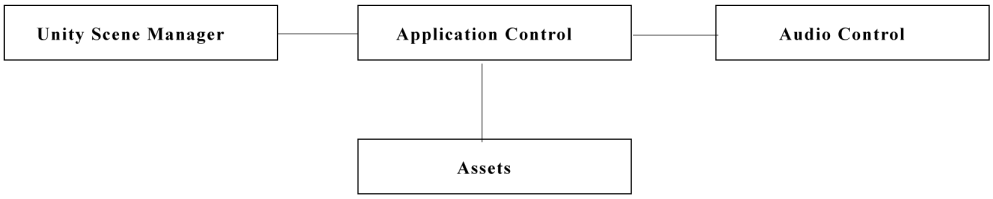
This section will split into two subsections as mentioned, a high-level discussing the game's base functionality along with its components and a design-level going over the game's class hierarchy.

4.1 High-Level Design (Architecture)

The high-level view or architecture consists of Facade's major components:

Utilizing Unity's engine, we are primarily using Unity's scene management system to load scenes for. The audio control is utilized by a script and serves as a component to the

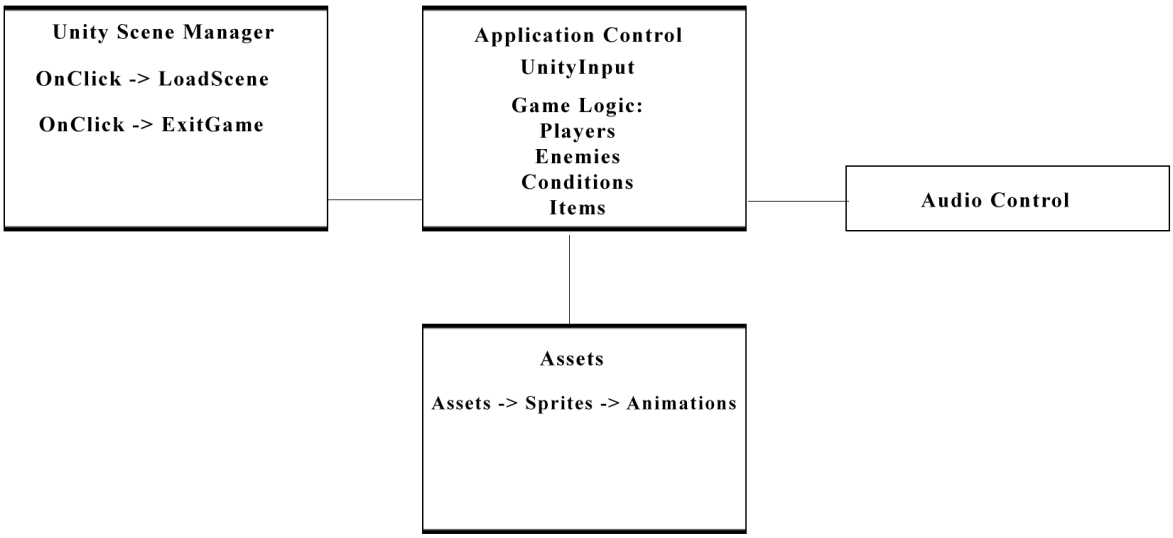
application along with the application using the assets provided to make the game whole.



- The **Unity Scene Manager** assists the application in managing the scenes throughout the game; such as menu and several levels.
- The **Assets** is a repository of free assets we found on the internet. Unity's engine takes advantage of them with the utilization of creating animations and sprite editing.
- The **Audio Control** manages the audio of the game. From our assets, it takes the audio file and plays the instructed audio as instructed.
- The **Application Control** is our container that holds the game together and manages all aspects of the application.

4.2 Mid-Level Design

The mid-level design goes over the basic functionality of the game, where we utilize Unity's engine for driving our application.



5 Use Case View

In this view, we will be highlighting the basic aspects of the game for each scene and what the user can do from each screen:

Main Menu:

1. Start Game: starts the game
2. About: displays information about the team and class
3. Exit Game: exits the application

Level 1-3:

1. How To Play UI: Tells the user how to play the game with the displayed graphic
2. Player proceeds through the level
3. Player defeats enemies for score count
4. Player retrieves item for health/mana heals
5. Player pauses game
    a. Resume: player returns to game
    b. Main Menu: player can return to main menu and quit
6. Player reaches end goal
    a. Next Level: player can proceed to next level
    b. Restart Level: player can restart level if they want to earn higher score
    c. Main Menu: player can return to main menu to exit
7. Player dies in the game
    a. Restart Level: player can restart level for another try
    b. Main Menu: player can return to the main menu and quit