Requirements and Analysis Document for Generic Platformer

Oscar Arvidson, Sam Salek, Anwarr Shiervani, Erik Wetter, Malte Åkvist

October 2, 2020
version 1.0

Contents

1 Introduction

The aim of this project is to create a 2D platformer game. It's a single-player game where the player can use a multitude of weapons to fight off different kinds of enemies. The game progress is built up using multiple levels and wave logic. This means when the current wave of enemies is defeated a new wave will appear.

The purpose of this project is to entertain our users and create stress relief in their everyday life. Generic platformer is supposed to create a fun and relaxing atmosphere, unlike competitive games how can form a very stressful environment at times. To achieve these goals and still build an enjoyable game to play the project used a cartoon-like design rather than a more realistic design.

1.1 Definitions, acronyms and abbreviations

Wordlist:

- Enemies:
- Waves:
- Levels:
- Platformer:
- Player:
- Block:
- Weapon:
- Hitbox:

2 Requirements

2.1 User Stories

Story Identifier: GPG001

Implemented: Yes

Story Name: Basic World

Description: As a player, I need to see a world in the game so that I have something to explore or interact with. A game world would let me see how I can interact when I'm in the game for example seeing where I can go.

Confirmation:
Functional:

- The player needs to be able to stand on some kind of ground.
- The game world should have 2-3 platforms.
- Platforms should have a collidable hitbox.

Non-functional:

- The game world's terrain should be built up using a grid.
- The game world will have three different texture tiles.

Story Identifier: GPG002

Implemented: Yes

Story Name: Player and Movement

Description: As a Player, I want to be able to see my character and to control my character by moving and jumping in order to make progress in the game.

Confirmation:
Functional:

- Have a movable player object/character.
- The players' character can move right by pressing D.
- The players' character can move left by pressing A.
- The players' character can jump by pressing W.

Non-functional:

Story Identifier: GPG003

Implemented: Yes

Story Name: Enemies

Description: As a Player, I want different types of enemies to appear in the game world and follow my character. Three different types will create a bigger variation for my gameplay and with enemies following me I will get a reason to move.

Confirmation:
Functional:

- Have three different types of enemies.
- Enemies will run towards players.
- If an enemy reaches the player, it will stop.
- The enemy will primarily follow the player in the x-direction.

Non-functional:

- Enemies should look different in order to tell them apart.

Story Identifier: GPG004

Implemented: No

Story Name: Player Shooting

Description: As a Player, I want to be able to shoot possible threats e.g enemies, in order to defend myself.

Confirmation:
Functional:

- The Player aims with the mouse.
- Player fires by clicking the Mouse1 button.
- When a player is shooting a projectile spawns.
- The weapon can be reloaded.
- The player can only shoot when the weapon has reloaded.

Non-functional:

Story Identifier: GPG005

Implemented: Yes

Story Name: Weapon Projectiles

Description: As a Player I want the projectiles fired from my weapon to move in the direction I am aiming to prevent enemies from coming too close to me.

Confirmation:
Functional:

- The projectile travels forward in the angle it was given.
- The projectile travels towards the designated target selected by the player.
- After the projectile has traveled for more than 3 seconds it is destroyed.
- The projectile collides with enemies and pushes them away.
- The projectile travels at a constant speed.

Non-functional:

Story Identifier: GPG006

Implemented: Yes

Story Name: Jump Collision Reset

Description: As a player, I want my character to regain the ability to jump when touching the ground so that I can jump more than one time during the game and not jump while I am in the air.

Confirmation:
Functional:

- Regain the ability to jump when touching the ground/platforms.
- Stop the player from being able to jump when in the air.
- Prevent the player to jump up and sit on the walls.
- Do not allow the player to climb walls by spamming jump.

Non-functional:

Story Identifier: GPG007

Implemented: Yes

Story Name: Collision Detection

Description: As a player, I want the game to react when my character shoots an enemy or when an enemy hits me, so that an enemy and a player can take damage/lose health.

Confirmation:
Functional:

- Collisions between the player and enemies should lower players' health.
- Collisions between enemies and projectiles should lower the enemy's health.
- Collisions between enemies and projectiles should remove projectiles.

Non-functional:

Story Identifier: GPG008

Implemented: No

Story Name: Basic Wave Manager

Description: As a player, I want to be able to fight waves of enemies so that there is a challenge and goal in the game.

Confirmation
Functional:

- A new wave will spawn x amount of different enemies (depending on which wave it is).
- Enemies will spawn at the edges of the map.
- The higher the wave, the more enemies will spawn.
- When the timer for each wave runs out a new wave will spawn.
- When all enemies in a wave are dead a new wave will spawn.

Non-functional:

2.2 Definition of Done

For a user story to be defined as done it has to meet all of its own acceptance criteria. All involved code and classes related to the user stories must be documented with java docs, in order to be understood by everyone associated with the project. The user stories acceptances will be tested through concrete test classes, human code evaluation, and user acceptance testing.

2.3 User interface

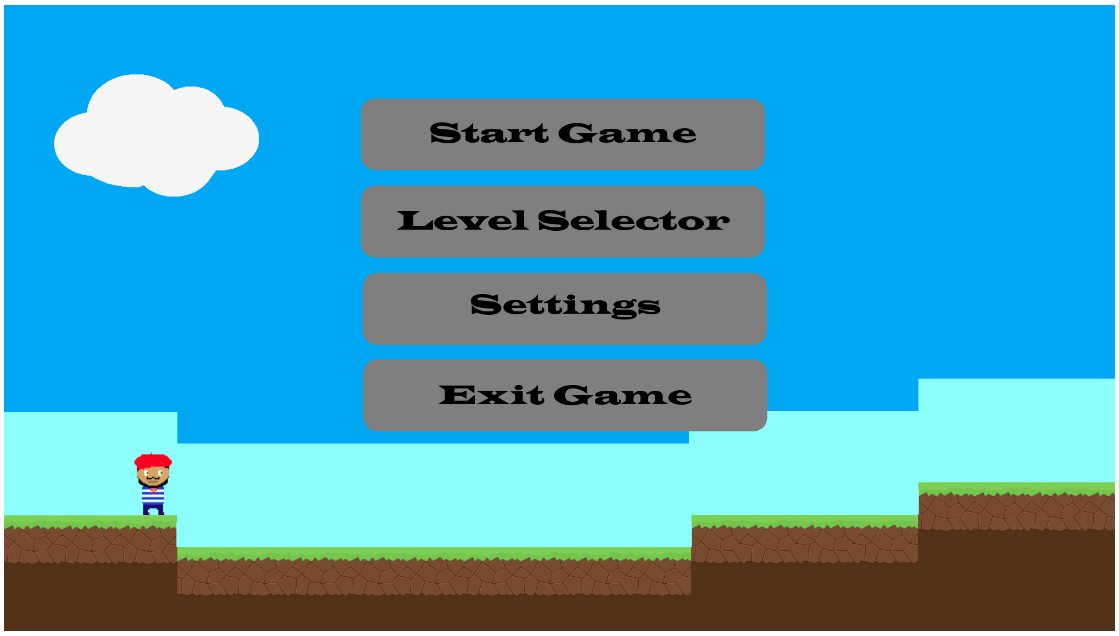

Figure 1: Sketch of main menu.

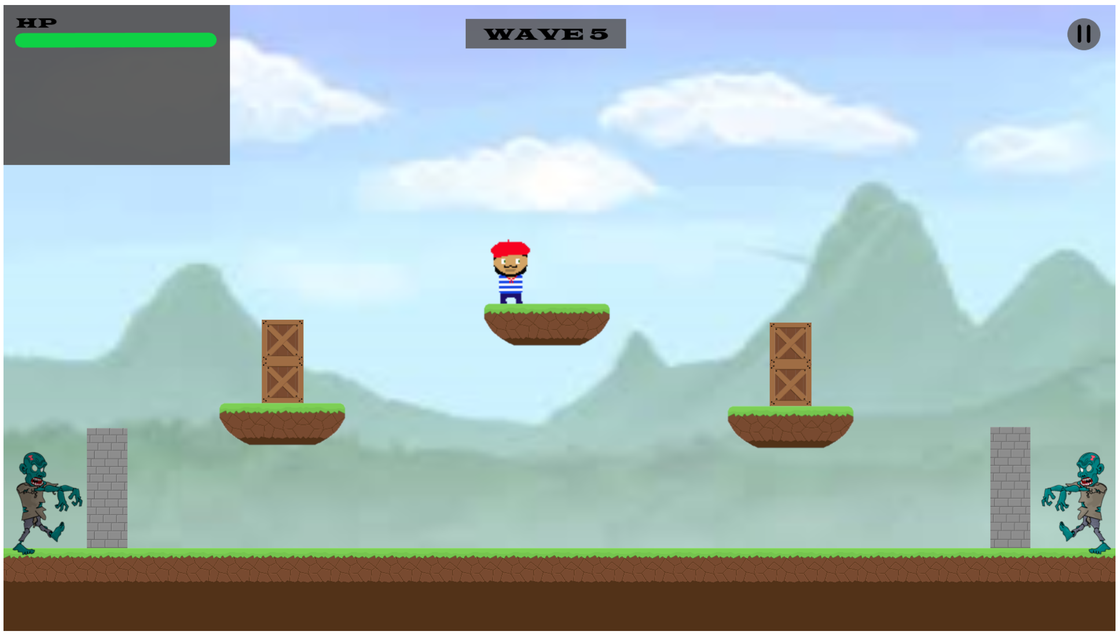

Figure 2: Sketch of in game view.

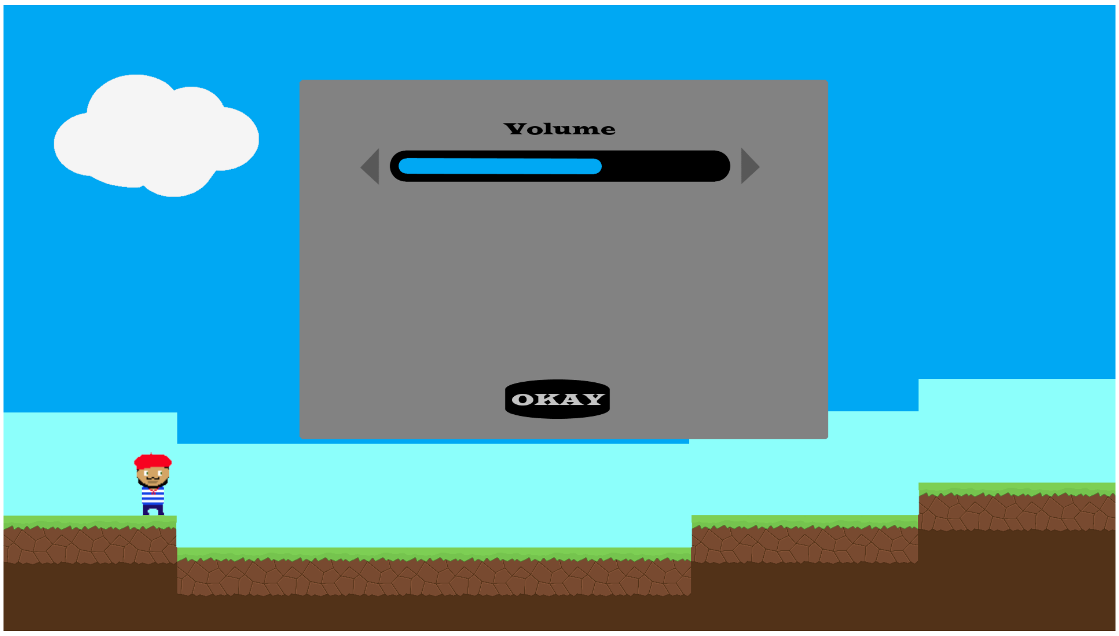

Figure 3: Sketch of settings menu.

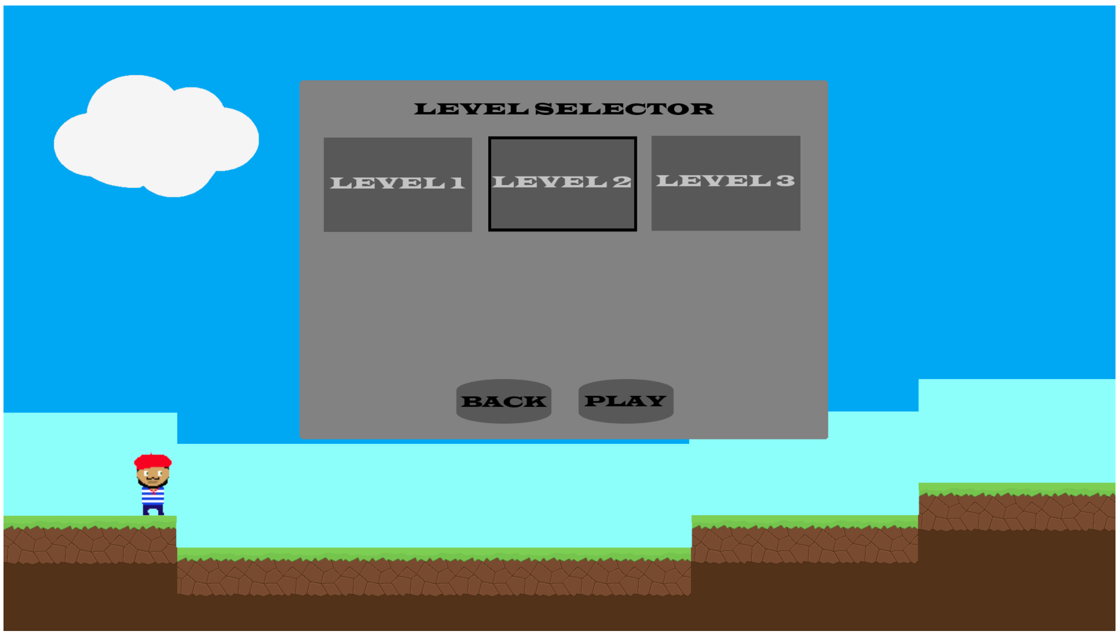

Figure 4: Sketch of level selector view.

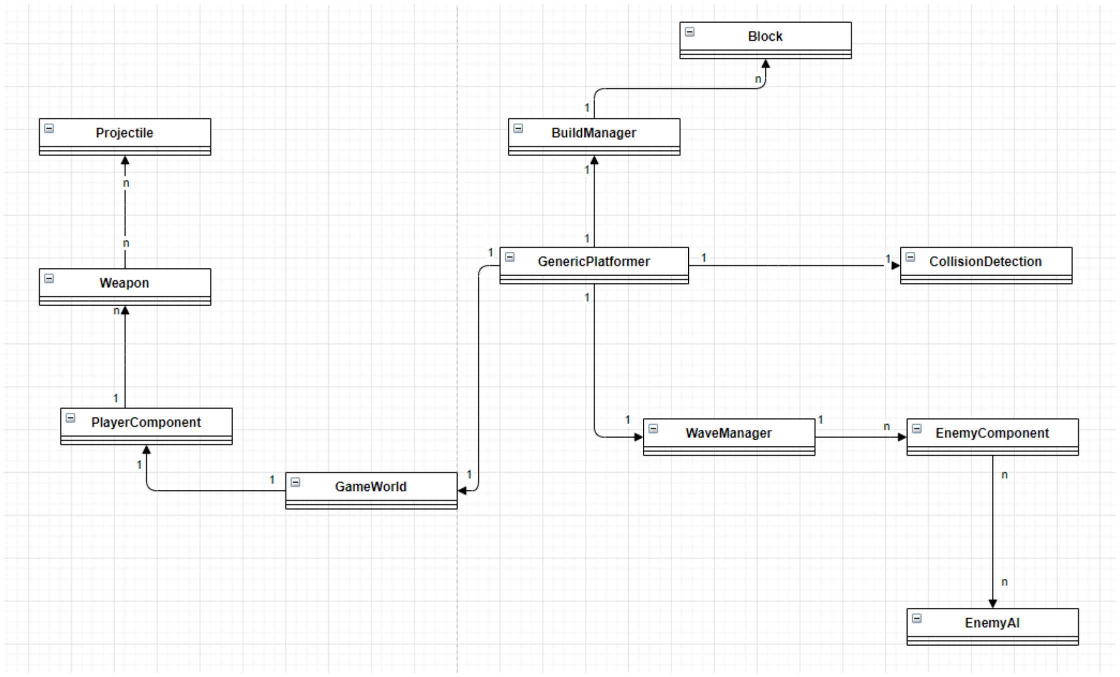

Figure 5: UML of the domain model.

3 Domain model

3.1 Class responsibilities

References

[1] A. Baimagambetov. (2020). Fxgl 11 wiki, [Online]. Available: `https://github.com/AlmasB/FXGL/wiki/FXGL-11` (visited on 09/30/2020).