Unnamed Platformer
for UTM CSCI 352

Colin Weatherly, Blade Johnson

Abstract

The project is a video game in the 2D platformer genre. The player is made to run through a level, and upon each successful completion, the level is changed to increase the difficulty. The intended target is people that play video games, primarily those that stream games as live reactions mesh well with high difficulties.

1. Introduction

This project is a video game in the genre of 2D platformer. The goal of the game is to reach the exit of the level that the player character is placed in. The level's layout will progressively get more complex with each clear, along with adding new hazards that increase the difficulty further. The penalty for failing a level is to be sent back to the beginning of said level, allowing the player to learn the level with every attempt.

The targeted audience for this project is people in the 18-34 age group who play video games on a regular basis. Specifically, those in the age group that livestream games. Due to the increasingly difficult challenge that the game provides, having live reactions of attempts to beat the game will provide content for the targeted audience.

1.1. Background

A 2D platformer is a game genre set on a 2D plane where the player controlled character must move and jump to avoid obstacles and reach the end.

We decided to make a platforming game because of our love for games in general and our shared interest in platformers. Because of this, we feel we can properly create one.

1.2. Impacts

This project is not meant to impact the world in some way. The point of a game is to provide entertainment, which will only impact a portion of the population if it were to gain traction within the community. At the current time, there is no plan for a meaningful story in the game, so the impact will be purely driven by the quality of the gameplay.

1.3. Challenges

The main challenges will most likely be coding the physics of the player character; developing new mechanics instead of being purely moving and jumping; keeping the game fun despite the difficulty.

2. Scope

This platforming game is considered done once three levels of the game are playable from start to finish with minimal bugs or exploits. The player should be able to get through these stages without too much unnecessary frustration with either the difficulty of these stages or the control of the player character. The player character should be fully implemented with all moves intact (moving, jumping) and all hazards(such as spikes) should fail the level upon contact with the player character. As a stretch goal, we want to include more levels with new hazards. An additional stretch goal is the addition of extra moves(i.e walljumping, sliding) the player character requires to progress in more difficult levels. An additional stretch goal is scaling the window and the game assets to fit more than one resolution.

2.1. Requirements

2.1.1. Functional.

- Player's completed levels stay completed – user's completed levels persist after the application is exited and started again

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Start New Game	User	Low	1
2	Level Select	User	Low	1
3	Character Jump	User	Medium	1

TABLE 1. USE CASE TABLE

- User inputs via designated keys are read – the application should register the inputs and move the player character accordingly
- Selecting New Game erases the user's save data – completed levels will return to being uncompleted, player is loaded into the first level as if starting the game for the first time
- Level fails upon contact with a hazard – should the player character run into a hazard(i.e spike), level is failed and character is returned to the starting point of the same level

2.1.2. Non-Functional.

- The game correctly displays in a 1280x720 resolution window
- The game runs at a smooth framerate, preferably 30 frames per second
- User's inputs have little to no delay, preferably under half a second

2.2. Use Cases

These are cases in which the user will interact with the program's user interface. The use cases can be seen in Table 1.

Use Case Number: 1
Use Case Name: Start New Game
Description: The player decides to begin the game from the beginning (Level 1). They will click on the "New Game" button (see Figure 2). This will load the first level of the game (see Figure 1). NOTE:This is only a conceptualized level meant to show loading into a level, not specifically level 1.
Process flow:

1) Player starts the program, which begins with the main menu loaded.
2) The player left-clicks the "New Game" option on the menu.
3) The game state is updated to gameplay, and the first level is loaded.

Termination Outcome: Gameplay has begun in level 1.

Use Case Number: 2
Use Case Name: Level Select
Description: The player is on the main menu. The player wishes to choose a specific level in the game to play. They will left-click the "Level Select" button on the main menu, and the screen will update to show a list of levels in vertical descending order. The player then left-click a button corresponding to a specific level, and the game will load the corresponding level (see Figure 3).
Process flow:

1) Player starts the program, which begins with the main menu loaded.
2) The player left-clicks the "Level Select" option on the main menu.
3) The level select screen is loaded, showing each level vertically in numerically descending order.
4) The player left-clicks the option corresponding to their desired level.
5) The game state is updated to gameplay, and the corresponding level loads.

Termination Outcome: Gameplay has begun in the player's selected level.

Use Case Number: 3
Use Case Name: Character Jump
Description: The player is in gameplay. They wish to make the player character jump (in order to avoid hazards such as spikes). The player then hits the space bar. The player character will then leap upward. The character will gradually rise up off the ground for a few seconds, then gradually fall back to the ground.
Process flow:

1) Player is in a stage of gameplay.
2) The player presses the space bar.
3) The player character then leaps off the ground into the air.
4) After a few seconds of gradually rising upward, the player character then gradually falls until hitting the ground.

Termination Outcome: The player character has completed a full jump.

Alternative: Player character is already jumping

1) The player presses the space bar as the player character is in mid-jump.
2) The input is ignored, as the game recognized that the character is already jumping.
3) The player cannot initiate another jump until the current jump has completed.

Termination Outcome: The player's space bar input is ignored while the character is already jumping.

2.3. Interface Mockups

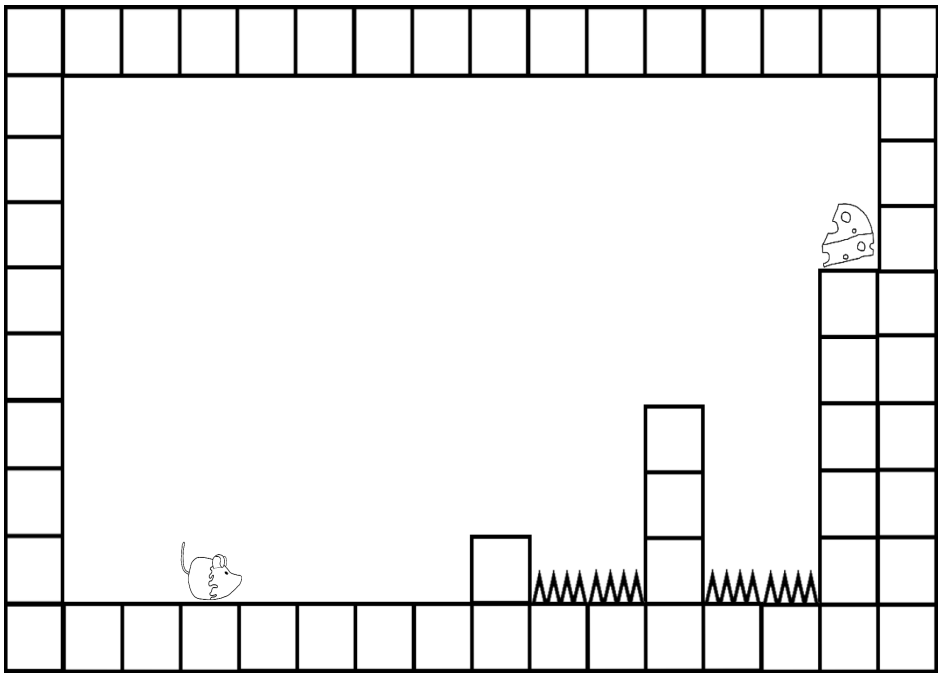

Figure 1. Mockup of a player loading into a level

3. Project Timeline

Go back to your notes and look up a typical project development life cycle for the Waterfall approach. How will you follow this life cycle over the remainder of this semester? This will usually involve a chart showing your proposed timeline, with specific milestones plotted out. Make sure you have deliverable dates from the course schedule listed, with a plan to meet them (NOTE: these are generally optimistic deadlines).

4. Project Structure

At first, this will be a little empty (it will need to be filled in by the time you turn in your final report). This is your chance to discuss all of your design decisions (consider this the README's big brother).

4.1. UML Outline

Show the full structure of your program. Make sure to keep on updating this section as your project evolves (you often start out with one plan, but end up modifying things as you move along). As a note, while Dia fails miserably at generating pdfs (probably my fault), I have had much success with png files. Make sure to wrap your images in a `figure` environment, and to reference with the `ref` command. For example, see Figure 4.

4.2. Design Patterns Used

Make sure to actually use at least 2 design patterns from this class. This is not normally part of such documentation, but largely just specific to this class – I want to see you use the patterns!

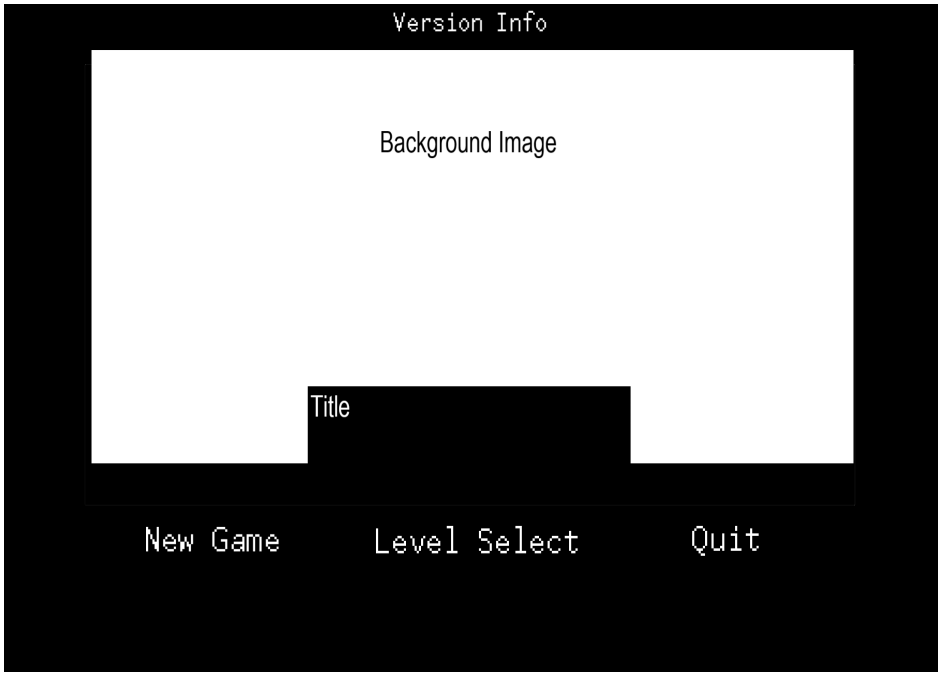

Figure 2. Mockup of the main menu screen

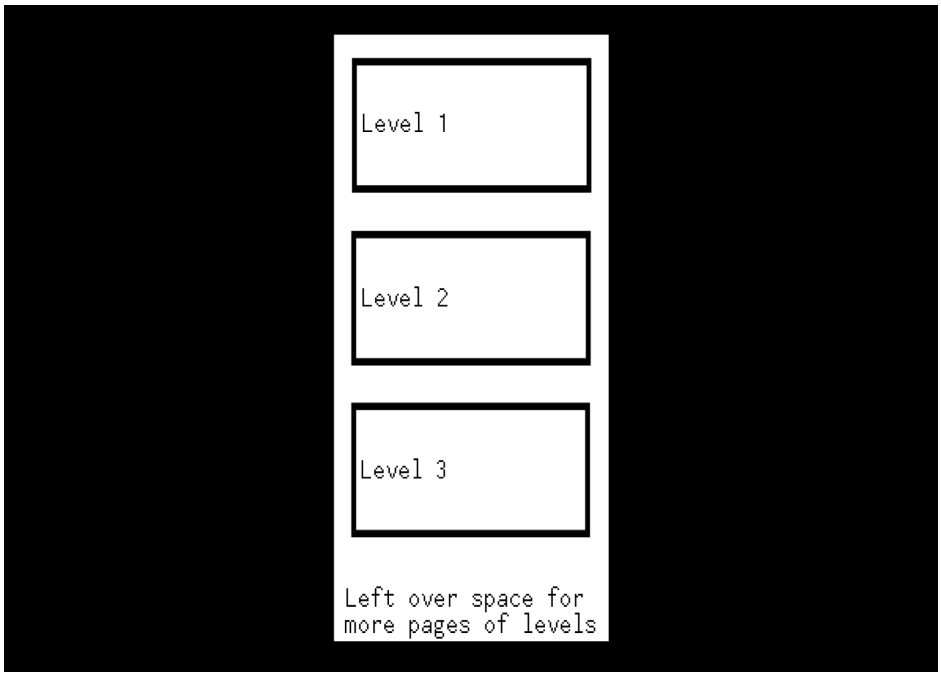

Figure 3. Mockup of the level select screen

5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.

Figure 4. Your figures should be in the *figure* environment, and have captions. Should also be of diagrams pertaining to your project, not random internet kittens

5.1. Future Work

Where are you going next with your project? For early deliverables, what are your next steps? (HINT: you will typically want to look back at your timeline and evaluate: did you meet your expected goals? Are you ahead of schedule? Did you decide to shift gears and implement a new feature?) By the end, what do you plan on doing with this project? Will you try to sell it? Set it on fire? Link to it on your resume and forget it exists?

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.