

# THE MIKU MEGA DOC

Certified Miku fans



Figure 1: Hatsune Miku (Fandom *Hatsune Miku*)

# Table of Contents

<b>Introduction</b>	3
<b>Who is Hatsune Miku?</b>	3
<b>Sprint 2</b>	4
- Division of Responsibilities	4
- Features	X
- Summary	X
- Drawbacks	X
- Results	X
<b>Sprint 3</b>	X
- Division of Responsibilities	X
- Features	X
- Summary	X
- Drawbacks	X
- Results	X
<b>Sprint 4</b>	X
- Division of Responsibilities	X
- Features	X
- Summary	X
- Drawbacks	X
- Results	X
<b>Sprint 5</b>	X

- Division of Responsibilities	X
- Features	X
- Summary	X
- Drawbacks	X
- Results	X
<b>Conclusion</b>	X
<b>Closing Statement</b>	X
<b>References</b>	X

# Introduction

The primary goal of the team Certified Miku Fans is to create a Zelda-like game starring Vocaloid's lead, Hatsune Miku. The team consists of Wilson Lin, Charlie Cavallaro, Vivian Qian, Timothy Blair, Noelle Lin, and Songyu Ye. The document will highlight the group's struggles and successes throughout the design and coding process, demonstrating the progression of the team's video game design capabilities and showcasing the group's ability to collaborate effectively, despite being an impromptu assembly of members.

## Who is Hatsune Miku?

[https://en.wikipedia.org/wiki/Hatsune\\_Miku](https://en.wikipedia.org/wiki/Hatsune_Miku)

(Insert Hatsune Miku here)

// In text citation

*(Who is Hatsune Miku?)*

# Meeting Log

9/10/2024 - Sprint 2 Division of Responsibilities discussion.

9/11/2024 (In class) - Sprint 2 sprite and state discussion.

\*Note from this point on In Class meetings are not recorded but assume they happened.

9/13/2024 - Recurring general discussion meeting.

9/23/25 - Enemy Sub-Team meeting to discuss and refine enemy functionality

9/23/2024- Grader Meeting 1 with Natalie

9/26/2024 - Grader Meeting 2 with Natalie

## Sprint 2

### Division of Responsibilities

- Player control (1 2 3) - Wilson and Charlie
  - Implement a Command for each movement when key is clicked
  - Implement a Command for each movement when key is held down
  - Implement a Command for each movement when key is released
- Blocks / Obstacle controls - TJ
  - Blocks appear on screen in a set position.
  - Can use T and Y to cycle through blocks.

- Item controls - Vivian
  - Keyboard and ICommand structure
  - Helped with Miku's early movement animation
  - Item interface and commands (cycling through items/summoning them)
  - Sprite Work (Photoshopped out all backgrounds and drew custom sprites)
- Enemy controls (2 commands, cycle enemies) - Songyu and Noelle
  - Finding + formatting sprites
  - Making the enemies move on their own.
  -
- Miku (Currently Link) controls (moving and throwing) - Wilson and Charlie
  - Miku moving and facing different directions
  - Attacking and moving
- Lead version control specialist (Github) - Charles Cavallaro
- Lead documenter - Wilson Lin

Alternate Separations:

Research:

- How to consolidate sprite classes especially animated or moving vs not animated.
- Later command patterns + other useful content covered later in the class.

By 9/14:

Link movement: Have movement working with as many abilities working.

Interface: Have a rough interface ready for use.

Other controls: Complete by 9/14

Blocks / Items: Have a single block or item with the cycling (Even if not all of them)

Enemies: Have at least one enemy ready with movement.

### [Interface Discussion](#)

**Folders:** All Interfaces in 1 folder, sort by interface implemented, all commands and controllers grouped.

### [Sprint 2 Code Review Notes](#)

### [Sprint 2 Reflection](#)

## Features

All of the features for this sprint were met, however there were drawbacks faced along the way. More documentation below will further explain the challenges.

Features included

1. WASD movement and direction change.
2. Miku's sword stab (In this sprint, we have it bound along with the other projectiles in the number keys)
3. E causes Miku to be damaged.
4. T and Y cycle between all the blocks showcasing the building blocks used to build the world.
5. U and I showcase all the items in the game and cycle through all of them.
6. O and P display the enemies with attack patterns and movement as they do in the game.
7. Q quits the game.

Features not included

1. R to reset the game.

## Summary

Sprint 2 showcased our individual commitment to completing assigned tasks; however, the group overlooked crucial steps of the design process. The most critical error during the DESIGN process was skipping the discussion phase. Our team, Certified Miku Fans, had forgotten to discuss and explore strategies, the team had eagerly jumped directly into implementation. This misstep resulted in a cascade of issues that were addressed with temporary band-aid fixes. A new foundation is currently being laid. While valuable lessons were learned, these band-aid fixes, although not born of ignorance to design patterns, were intended to be used as stepping stones to test features, however if they were removed it would cause the entire project to collapse entirely.

## Drawbacks

- Flawed foundational design.
- Poor planning.
- Limited discussion between different task holders. (Causing varied ways of implementation)
- Extensive refactoring required.

## Results

The results of the project were neutral. Although a weak code architecture was created, the project has high functionality. Overall the group had key takeaways that will be transitioned into the next sprints.



## Sprint 3

### **Objectives:**

- Continue to develop more core features of your 2D game framework.
- Implement collision handling for all types of collisions that can occur, causing state transitions or position changes when necessary.
- Instead of placing all objects on screen at once, create the individual "rooms" (or "screens" if you prefer) of the dungeon each with its own subset of objects. Store this information in a file (recommendation: csv or xml) and write code to initialize objects based on the file's contents.
- In addition to the regular dungeon, create an artificial level that contains an instance of Link and all types of objects that are found in your first dungeon. For testing purposes, include a way for the user to quickly switch to any room in the dungeon instead of having to walk through multiple rooms to reach it.
  - **Zelda:** This means all of the rooms in the first level.

## Division of Responsibilities

### **Backlog:**

Refactoring code to follow disciplines: Charlie and Wilson

### **Assignments:**

Collision: Vivian, Songyu, and Noelle

Level loading: Wilson, Charlie, and TJ

Task	Date started	Functional date	Final completion date
Refactoring code	9/30/24	10/13/24	10/18/24
Collision	9/30/24	10/15/24	10/18/24
Level Loading	9/30/24	10/15/24	10/18/24
Merging	10/16/24	10/17/24	10/17/24

Task tracking was done within discord.

Code reviews and README can be found on the most recent branch on GitHub.

<https://github.com/Crcav926/LegendOfZelda>

## Features

All of the features for this sprint were met, however there were drawbacks faced along the way. More documentation below will further explain the challenges.

Features included

- Player collision with blocks and entities.
- Enemy collision with blocks and entities.
- Loading of entities within rooms.
- Travel between rooms.
- Refactored framework.
- Resolution Independence. (Changed within Constants.cs)

# Summary

Sprint 3 for our team, Certified Miku Fans, was characterized by a division into two groups, focusing on distinct, but critical, functionalities of collision and level loading. The division was intended to streamline our efforts and understanding of having specialists at each task, however the growing pains arrived when it was time to merge codebases for collision and level loading. Several refactors were then made to remedy the merging of code. Additionally, the sprint saw the implementation of lesser features like a FPS counter, resolution independence, and a Constants file to hold global variables. Although development was halted due to fragmented development, the group produced a product that contains collision and level loading.

# Drawbacks

Drawbacks faced:

- Merging struggles
- Resolution Independence (Having set variables that do not scale)
- Fragmented development
- Moving magic numbers (More to be moved in future sprints)
- Refactoring previous work.
- Swapping between levels

# Results

The results of the project were positive. Certified Miku Fan's refactored numerous elements of the codebase, which has improved through adding many design patterns as previously learned. Numerous tasks were completed throughout Sprint 3. Collision and level loading were successful and functional. Although there are still items on the backlog, the team had key positive takeaways that will be transitioned into the next sprints.

## Sprint 4

### Objectives:

## Division of Responsibilities

[https://docs.google.com/document/d/1TYSq-qvviGFkio\\_t9N\\_4F9mgEgRr3p83fUWjA-JLXyM/edit?tab=t.0](https://docs.google.com/document/d/1TYSq-qvviGFkio_t9N_4F9mgEgRr3p83fUWjA-JLXyM/edit?tab=t.0)

## Features

Not all of the features for this sprint were met, and there were drawbacks faced along the way. More documentation below will further explain the challenges.

### Features included

- HUD implementation (HP, Coin counters, weapons equipped, map)
- Locked doors
- Camera transitions
- Death animation

### Backlog / Known Bugs

- Need to add full map implementations
- Pause screen inventory (Currently uses the key '3' to swap between inventory items)
- Rooms 7,8,13 don't collide for some strange reason. Specifically rooms to the left and above the first room.

- The reset after the death has not been implemented yet. The game just permanently stays on the death animation
- Old man's room needs to be completed.
- Previous backlog tasks are not finished.

## Summary

Sprint 4 for our group showed the difficulties of shotgun surgery. We had many tasks completed at a reasonable time, however we realized after merging and implementing them together, it would break other parts of code. For example, fixing our camera transitions would break any room to the left of the main room as it's considered “negative” position variables and we only check for collisions with positive values. This was promptly fixed, but a main issue was having locked doors functioning, and after merging code many other parts of code would break along the way. The team had great pacing with tasks this sprint, but fell short as they faced the wrath of shotgun surgery.

## Drawbacks

Drawbacks faced:

- Merging struggles
- Camera transitions (Would move everything on the screen including the hud in a weird way)
- Fragmented development
- Refactoring previous work.

## Results

The results of this project resulted in the major bugs being fixed, however in doing this, many smaller bugs sprouted within the code and will be refactored and fixed in the next sprint.



## References

*Hatsune Miku. Fandom*, [https://vocaloid.fandom.com/wiki/Hatsune\\_Miku](https://vocaloid.fandom.com/wiki/Hatsune_Miku). Accessed Sept. 2024.

“Who Is Hatsune Miku?” *About HATSUNE MIKU | CRYPTON FUTURE MEDIA*,  
ec.crypton.co.jp/pages/prod/virtualsinger/cv01\_us. Accessed Sept. 2024.