CS 419
Fall 2016

Hula: Project Plan

"Hula" Team Members
Jeannine Amm
Raymond Lo
Aaron Sealy

Introduction

The Hula Team will be completing a 2D Unity-based game similar to the original series of Super Mario Bros. games. As a team, our experience level with Unity and C# ranges from none to minimal, but this project choice enables us to develop some skills in a software suite that is fairly dominant in the game development industry, as well as to gain confidence in a major programming language.

User Perspective

From an end user perspective, games such as ours provide entertainment while also challenging the brain to find a solution to each level. This game will have 4 different challenges in the form of 3 levels and a final boss level. The target user will have already played some version of Super Mario Bros. and, as such, the user will have a familiarity with the game going in, as well as a certain level of expectations for both the appearance of the game as well as its style of gameplay.

Client

The instructor, Benjamin Brewster, will be considered the client. The client's requirements will be the requirements outlined in the project proposal by Aaron Sealy, which were approved by the instructor, as well as the requirements outlined in the class syllabus.

Client Requirements

- The creation of a 2D / side-scrolling Super Mario-like game.
- It must be created using the Unity Platform with C# as the scripting language.
- There must be 3 short levels as well as a "boss room."
- Each level and the boss room will be capable of being finished in less than 30 seconds and each level will have a 30-second countdown displayed.
- The 3 levels will include an "earth"-type, a "sky"-type, and a "fire"-type level. The boss room will be at the end of the fire level.
- The "earth" level will take place above ground and will include enemies, structures to climb, and hidden items.
- The "sky" level will take place at cloud level and the player will move across objects that include platforms that fall when touched. There will also be falling objects that can injure or kill the player.
- The "fire" level will take place in a dungeon maze. It will feature lava pits. The boss room will be at the end of this level.

- The only requirement for the boss room is that it restarts the countdown.

Sources

- We will use fan-made sprites and images readily available on the internet, referencing each source.
- Fair use: This software is for educational purposes only and has no commercial aspects to it and it is thereby able to make limited use of Nintendo's imagery and characters under the "fair use" provisions of the copyright act. All sources will be credited.

Initial Plans

Scenes/Levels (in order):

- Main Menu
    - Start Game Option
        - Load Game (if game file exists)
        - New Game
    - Exit Option
        - Confirm choice to quit
    - Other Options Button (available from any level)
        - Instructions
        - Mute
        - Save and Quit (with confirmation check)
- Earth Level
- Sky Level
- Fire Level
- Boss Level
- High Scores
    - Loads and displays from high score file
- "Win" Screen / End Credits

Global / Reuse Game Objects:

Mario:

- Scripts
- Sounds
- Actions / Animations
    - Walk
    - Run
    - Grow Big
    - Jump
    - Shoot Fireball
    - Change Color
    - Duck

    - Die

Items:

- Bricks
    - Shimmer animation
    - Breaking animation and sound when destroyed by headbuts
    - Some have hidden items, such as mushrooms, when broken
- Clouds
    - They are different sizes and types of clouds
    - Small stationary clouds - these can be walked on and do not fall
    - Small and large moving background clouds - no player interaction, move slowly across the screen from right to left.
    - Cloud platforms - These can be walked on but fall after a couple seconds
- Pipes
    - Houses Piranha Plant (see below)
    - Some pipes are transport pipes
    - When Mario ducks on a transport pipe, Mario enters the pipe and disappears
    - Sound effect and animation on player entering or leaving pipe
- Coins
    - Spinning Animation
    - Disappears on collide
    - Adds 1 to coins
    - Has sound effect
- Grow Big Mushroom
    - Increases Mario size if not already big
    - Adds one hit point if hit points are less than 2
    - Has sound effect
- Flower Power
    - Increases Mario size if not already big and turns him orange
    - Adds one hit point if hit points are less than 2
    - Activates ability to throw fireball
    - Has sound effect
- Fireball
    - Can be thrown if Mario has received flower power
    - Kills Goombas and Piranha Plants and knocks Koopa Turtles upside down
    - Has sound effect
- 1-Up Mushroom
    - Increases the number of lives by one
    - Has sound effect

Enemies:

- Goomba (Walking mushroom)
    - left, right, bottom collision removes one hit point from player

    - Top collision kills mushroom
    - Has walking animation
    - Has sound effect
    - Has basic AI (follows player)
- Flying Goomba
    - Same as Goomba, but with flight and flying animation
- Koopa Troopa (Turtle)
    - Walking state
        - left, right, bottom collision removes one hit point from player
        - Top collision puts turtle in hiding
    - Hiding in shell
        - Resumes walking state if not touched for 5 seconds
        - Contact with the player sends the shell flying off in the opposite direction. If it hits something, then it bounces back.
    - Has animations for walking, hit, hiding,
    - Has sound effect
    - Walks back and forth in a given patrol zone
- Piranha plant
    - Hides in pipe opening until Mario enters a trigger zone
    - Rises out of upright pipe opening
    - Has moving animation
    - Has sound effect
    - Any collision with player removes one hit point from player
    - Can only be killed by a fireball
    - Returns to hiding state after 4 seconds

Game Manager:
- Each level inherits from parent class
- Contains
    - Health Points
    - Current score
    - Music manager
    - Score manager
    - Item accumulation (e.g. count of coins)
    - Persistence Script
        - Player Object
        - HighScore Object
        - Save Player File
        - Load Player File
        - Save HighScore File
        - Load High Score File

Audio Manager:

- Manages different types of sounds
- Adjusts music level to accommodate sound effects

Enemy Animation Controller:

- Different enemies can inherit from one controller
- Koopa, Goomba, and Bowser all have similar basic animation triggers
- Specific traits can be added to the inherited controller

Game Persistence Information:

Possible variables (others possible):

- earthStatus (0 for not finished level, 1 for finished level)
- earthTime (time in seconds to complete level 1)
- skyStatus (0 for not finished level, 1 for finished level)
- skyTime (time in seconds to complete level 2)
- lavaStatus (0 for not finished level, 1 for finished level)
- lavaTime (time in seconds to complete level 3)
- bossStatus (0 for not finished boss, 1 for finished boss)
- bossTime (time in seconds to complete boss)
- coins (count of coins earned)
- HitPoints (starts at 1, sets to 2 if big or firepower, 0 if dead)
- isbig (0 if no, 1 if yes)
- isFire (0 if no, 1 if yes)
- lives (starts at 0, increments on life mushroom accumulation, decrements on death)
- bosshealth (percentage multiplier? 0,1,2,3,4,5)
- enemyDef (count of enemies defeated)
- cumulativeScore (score calculated at end of each level and added to this value)

File for GameSave

- Saves preferences above
- Saves top ten high scores (quickest times or most coins?)
- Save status
- Load status
- Present option to save at end of each level.

File for High Score Table

- Top ten based on calculation
- Score Calculation: (Coins x coins multiplier) + (Enemies defeated x enemy multiplier) + ((30 - earthTime) x seconds multiplier) + ((30 - skyTime) x seconds multiplier) + ((30 - lavaTime) x seconds multiplier) + ((30 - bossTime) x seconds multiplier) + (ifBig x big Multiplier) + (hiddenItems x hidden Item Multiplier)

Folder Structure:

- Animation - animations and animation controllers
- Audio - sound files
- Font - text fonts
- Materials - templates to describe shading / color etc that can be added to an object.
- Physics Materials - physics description of an object e.g. friction, bounciness
- Prefabs - templates / groupings for layout items and game objects
- Scenes - different level / menu layouts
- Scripts - C# scripts for game
- Sprites - 2D images
- Textures - non sprite images

Game Objects Unique to Each Level:
Unique Backgrounds
Unique Wall / Floors
Special Items (such as falling platforms, falling boulders, etc)
2-3 Unique enemies or threats (such as Thwomp, flame throwers, etc)
Unique Environmental threats (such as lava, falling to death, etc)

Level Testing

- Individual Level Testing
- Integration Testing

Software

- The Unity game development engine
- C# for scripting

Minimum Requirements to Run Finished Game[1]:

- OS: Windows XP SP2+
- Graphics card: DX9 (shader model 3.0) or DX11 with feature level 9.3 capabilities.
- CPU: SSE2 instruction set support.

Team Member Tasks

Aaron Sealy

Task	**Time Estimate (hrs)**
Week 3 - Work on global items Work on global enemies Loading and saving data, Continue to plan Lava level Submit video update	16

Week 4 - Complete global items and enemies Complete persistence data Complete Lava Level planning Start on background/floors/walls for Lava Level	14
Week 5 - Player / Enemy programming unique to Lava Level Start Collision interactions for Lava Level	13
Week 6 - Player / Enemy Programming unique to Lava level Complete collision interactions for Lava Level Produce mid-point check materials	17
Week 7 - Complete Rough Lava Level Lava Level Testing Start Boss Level (as a team)	15
Week 8 - Finish Boss Level (as a team) Boss Level Testing	10
Week 9 - Finish level testing, Integration Testing Categorize and Fix high priority bugs and updates from testing	12
Week10 - Final Report Input Demo File Submission	8
Total Time	**105**

Raymond Lo

Task	**Time Estimate (hrs)**
Week 3 Work on global items Work on global enemies Integrate persistance Work on saving and loading data Planning of Ground level	15
Week 4 Complete global items Complete global enemies Complete persistence data Complete Ground level planning Start tunnel actions for Ground level Start player/enemy interactions Ground level	17
Week 5	14

Player / Enemy programming unique to Ground level Start Collision interactions for Ground level Submit video update	
Week 6 Player / Enemy programming unique to Ground level Complete Collision interactions for Ground level	14
Week 7 Complete rough Ground level Ground level Testing Start Boss Level (as a team)	15
Week 8 Finish Boss Level (as a team) Boss Level Testing	10
Week 9 Finish level testing, Integration Testing Categorize and Fix high priority bugs and updates from testing Submit video update	12
Week 10 Final Report Input Demo File Submission	8
Total Time	**105**

Jeannine Amm

Task	**Time Estimate (hrs)**
Week 3 - Begin Global enemy prefabs Begin Global items Integrate Persistence, Loading and Saving Data, High score table generation	15
Week 4 - Complete global prefabs, Complete persistence data Complete Sky Level layout, Start platform actions for Sky Level Start player/enemy interactions for Sky Level Submit video update	17

Week 5 - Player / Enemy programming unique to Sky level Collision interactions for Sky Level	15
Week 6 - Player / Enemy Programming unique to Sky level Collision interactions for Sky Level	15
Week 7 - Finish Player / Enemy programming unique to Sky level Collision interactions for Sky Level Complete Individual Level Testing Start Boss Level (as a team)	15
Week 8 - Finish Boss Level (as a team) Boss Level Testing	10
Week 9 - Finish level testing, Integration Testing Categorize and Fix high priority bugs and updates from testing Submit video update	12
Week10 - Final Report Input Demo File Submission	6
Total Time	**105**

Task Delegation

Report	***Due***	***Format***	***Assignee to Submit***
Project Plan	Oct 3rd	Written Plan	[All]
Update Week 3	Oct 10th	Video	Aaron
Update Week 4	Oct 17th	Video	Jeannine
Update Week 5	Oct 24th	Video	Raymond
Mid-Point Check	Nov 4th	All files + written report	Aaron
Update Week 8	Nov 14th	Video	Jeannine
Update Week 9	Nov 21st	Video	Raymond
Final Report	Dec 2nd	Written (including instructions to run)	[All]
Demonstration	Dec 2nd	All files zipped	[All]

Graphical Examples

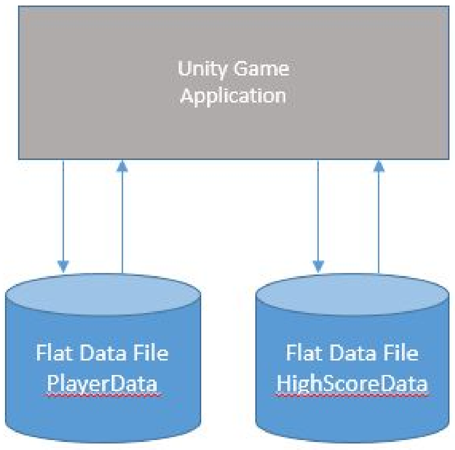


Figure 1.1 Game Architecture

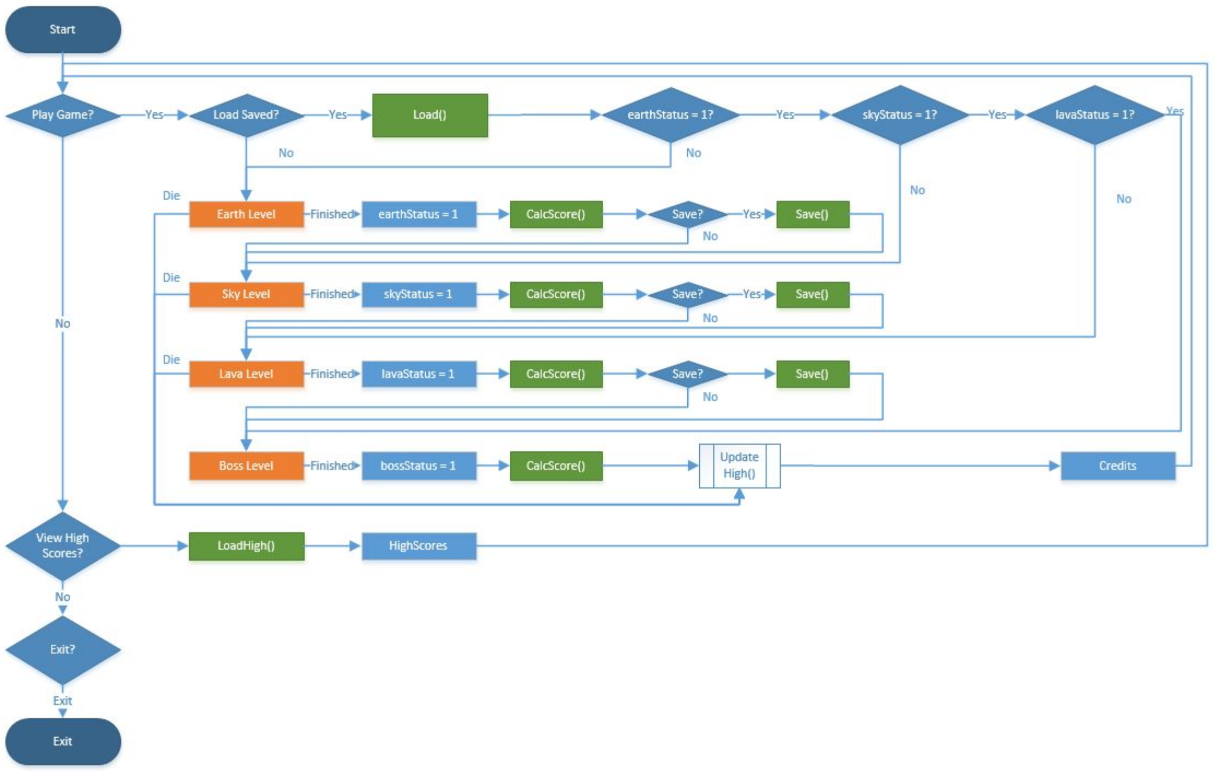


Figure 1.2 Game Flow

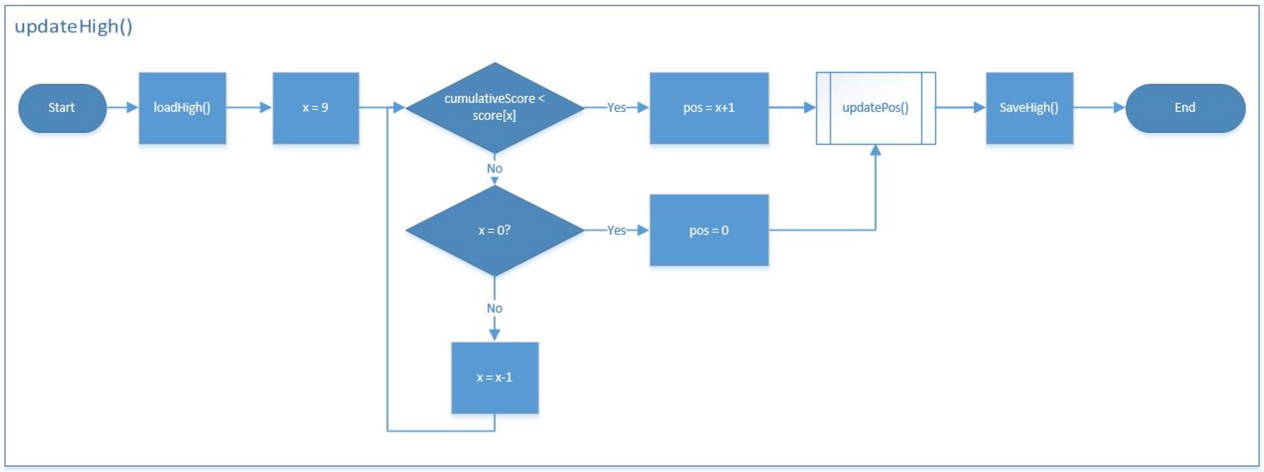
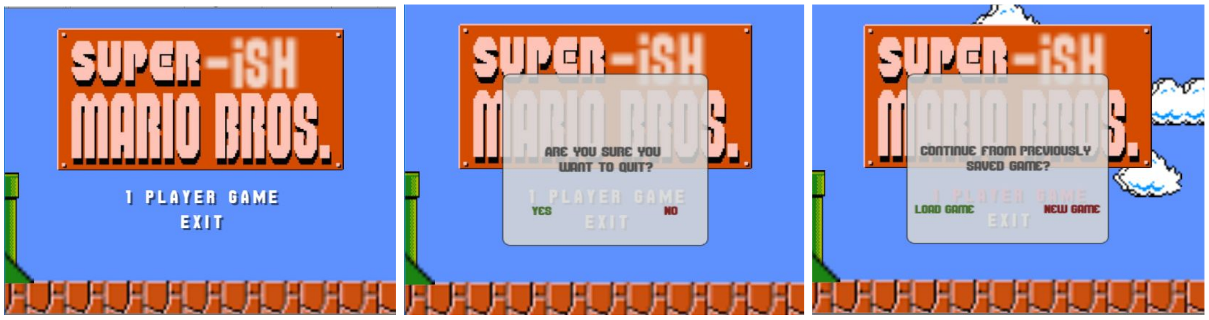


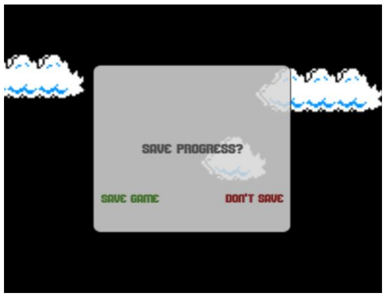
Figure 1.3 High Score Tables update flow

Levels:

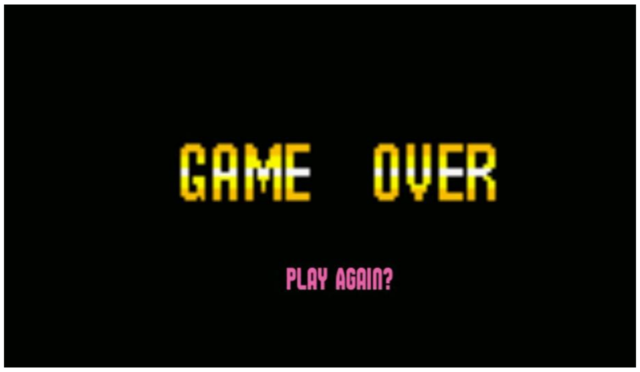
Menu Draft


SUPER-iSH
MARIO BROS.
1 PLAYER GAME
EXIT
ARE YOU SURE YOU WANT TO QUIT?
YES
NO
CONTINUE FROM PREVIOUSLY SAVED GAME?
LOAD GAME
NEW GAME

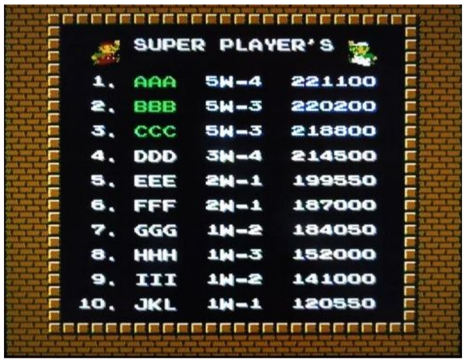
Save Game Between Levels Draft:



Game Over



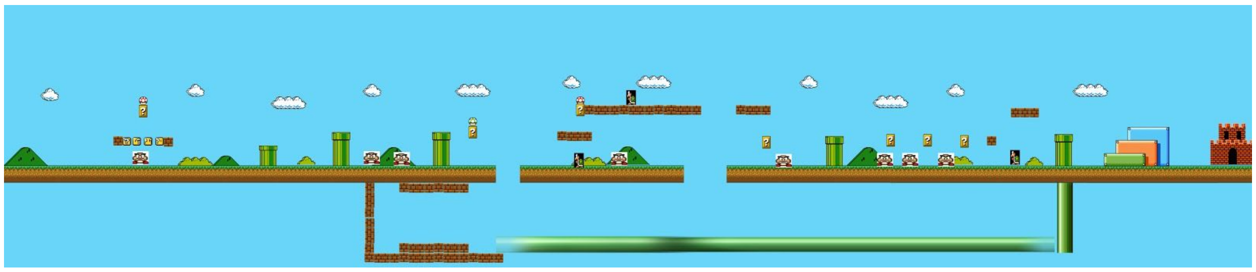
High Score Table Example (Similar to):



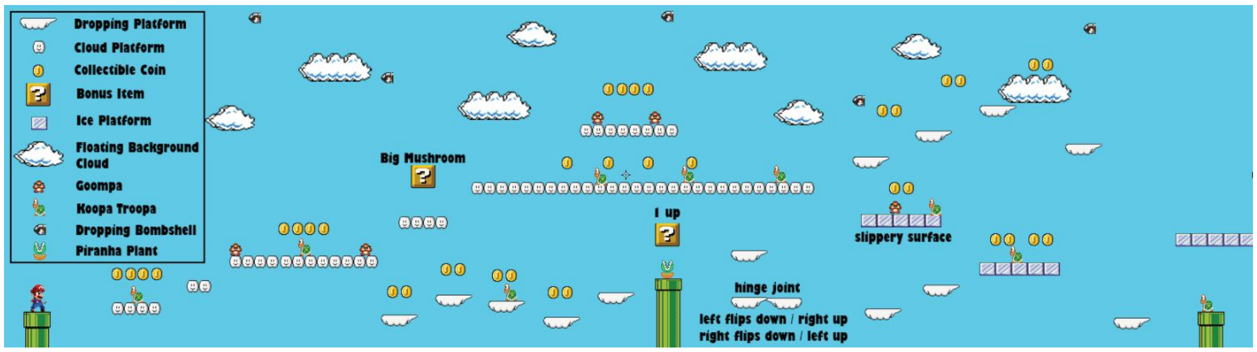
SUPER PLAYER'S

1.	AAA	5W-4	221100
2.	BBB	5W-3	220200
3.	CCC	5W-3	218800
4.	DDD	3W-4	214500
5.	EEE	2W-1	199550
6.	FFF	2W-1	187000
7.	GGG	1W-2	184050
8.	HHH	1W-3	152000
9.	III	1W-2	141000
10.	JKL	1W-1	120550

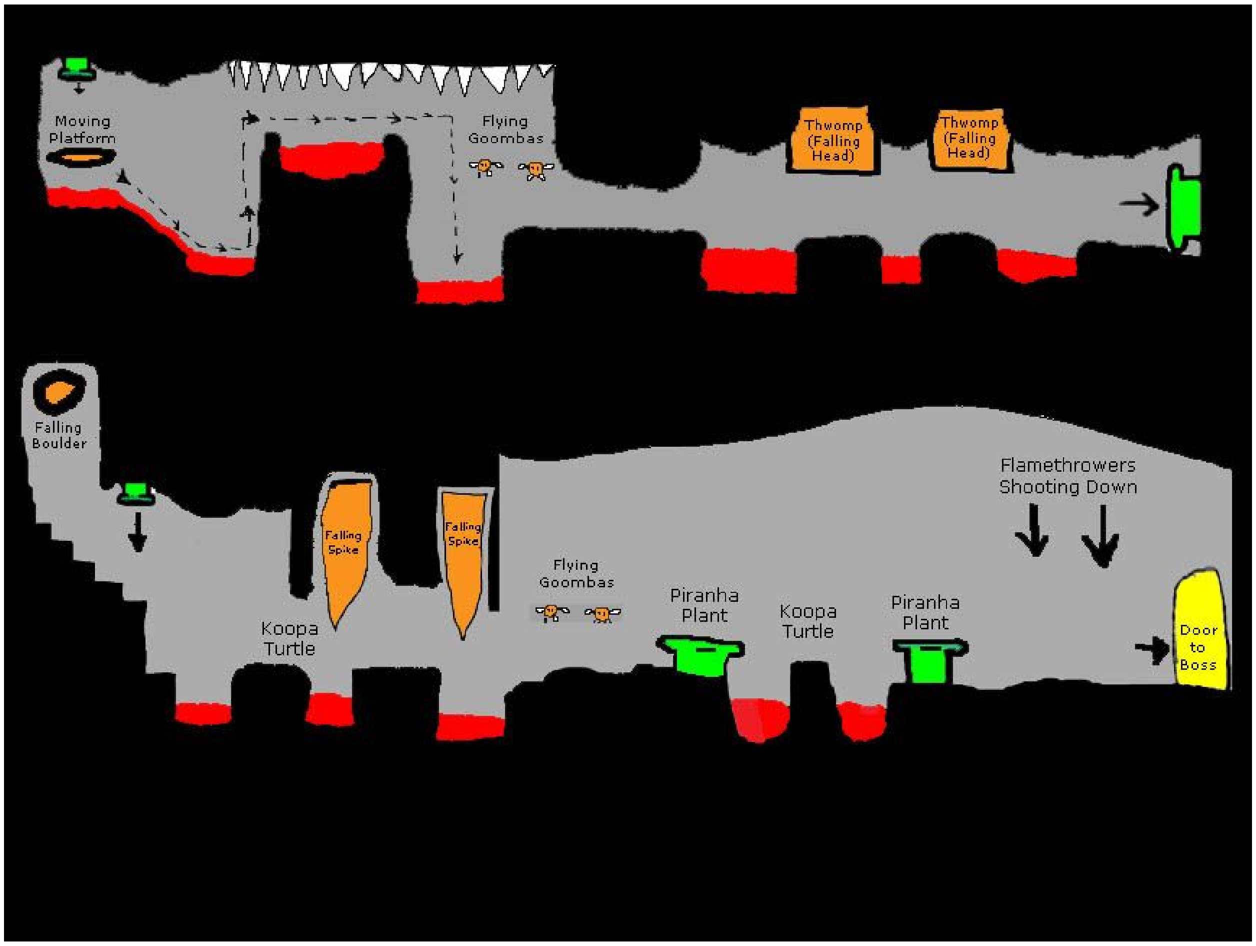
Ground Level Draft:



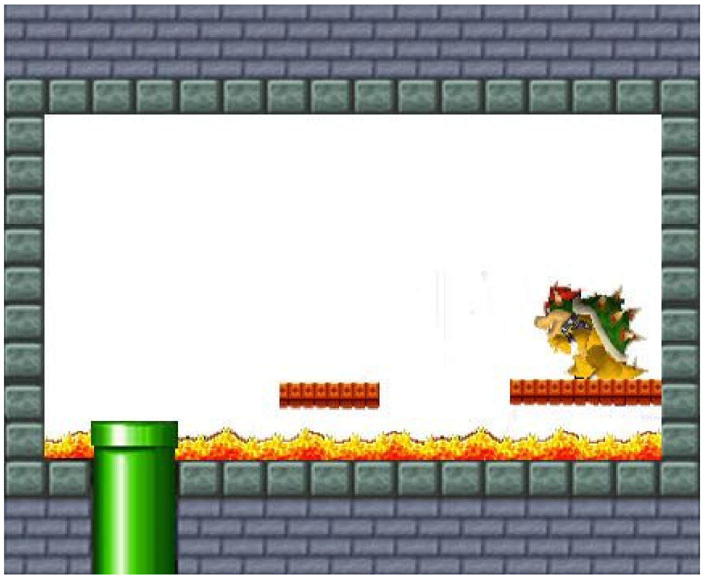
Sky Level Draft:



Lava Level Draft



Boss Room Draft



Conclusion

The Hula team hopes to create a Super Mario Bros.-like game that recalls the imagery and gameplay of the original Super Mario games by making use of the features of the Unity game engine, managed through scripting in the c# language. The project should take no less than 300 hours to complete and should be completed on schedule.

References

[1] https://unity3d.com/unity/system-requirements
[2] https://unity3d.com/learn/tutorials/topics/scripting/persistence-saving-and-loading-data
[3] https://unity3d.com/learn/tutorials/topics/scripting/persistence-saving-and-loading-data