



UNIVERSITI TUNKU ABDUL RAHMAN (UTAR)

Lee Kong Chian Faculty of Engineering & Science (FES LKC)

**Bachelor of (Honours) Software Engineering [SE]/ Applied Mathematics with
Computing [AM]**

Project [*Medieval*]

Group Name	Medieval
Student Names & (IDs)	Lu Jason 1906449 Cha Hong Yip 1804295 Cheah Ho Ching 1801494 Kelvin Chang Yik Yang 1801255
Due Date	16/9/2022
Lecturer	Tai Liang Kwang
Marks	

Table of Contents

Declaration	4
Project Overview	8
Game Overview (Written by Cha Hong Yip)	8
Game Premise (Written by Cheah Ho Ching)	9
Prominent game features (Written by Kelvin Chang Yik Yang)	10
Team Designation	11
Game Engine Details	12
Game Engine Version and Summary (Written by Lu Jason)	12
Game Engine Features (Written by Cha Hong Yip)	14
Scripting/ Programming Language (Written by Cheah Ho Ching)	16
Game Engine Technical Issues and SetUps (Written by Lu Jason)	18
Hardware Technical Specifications & Requirements (Written by Kelvin Chang)	25
Project Technical Design Overview	26
Game Technical Specifications (Written by Cha Hong Yip)	26
Project Game Development Setup	27
Project Organization Overview (Written by Lu Jason)	27
Game Flow/ Scenes Overview (Written by Lu Jason)	29
Programming Conventions and Designs	34
Overall Script Setup	34
Coding Conventions and Organization	35
Naming Conventions (Written by Lu Jason)	35
Feature Implementation Summary	35
Game Object Design Setup (Written by All)	35
Player Controls Implementation Overview (Written by Lu Jason)	42
Game Physics Implementation Overview	45
Jump Physics and Implementation (Written by Lu Jason)	45
Animation Implementation Overview (Written by All)	46
Audio Implementation Overview	62
Audio File Setup (Written by Lu Jason)	62
Sound Effects Overview (<i>Written by Lu Jason</i>)	63
Background Track Overview (<i>Written by Lu Jason</i>)	64
Game AI Overview [<i>script</i>]	64
Dragon	64

Phoenix	65
Eagle	65
Particles Overview	66
Particle System (Written by Lu Jason)	66
Game Interactable Objects Design and Implementation (Written by All)	68
Game Interface Setup and Implementation	70
Camera[s] Setup and Implementation (Written by Lu Jason)	70
Canvas and other GUI Setups and Implementations (Written by Lu Jason)	70
Game Options Setup and Implementation (Written by Lu Jason)	72
Project Development Progression	74
Project Schedule (Written by Lu Jason)	74
Progress Report (Written by Lu Jason)	75
References [<i>in APA format</i>]	75

Declaration

I, ___Lu Jason___ (Name), Student ID No. _1906449_ hereby solemnly and sincerely declare and confirm that I have read, understood, and shall abide and comply with all laws, rules, regulations, guidelines and lawful instruction of the University and its staff in relation to the commencement of any assessment / examination during my programme of study in Universiti Tunku Abdul Rahman.

I hereby declare that my submission for all assessment / examination during my programme of study in the University shall be based on my original work, not plagiarized from any source(s) except for citations and quotations which have been duly acknowledged. I am fully aware that students who are suspected of violating this pledge are liable to be referred to the Examination Disciplinary Committee of the University.

Signature : 

Name : Lu Jason

ID No. : 1906449

Date : 9/9/2022

I, ___Cha Hong Yip___ (Name), Student ID No. _1804295_ hereby solemnly and sincerely declare and confirm that I have read, understood, and shall abide and comply with all laws, rules, regulations, guidelines and lawful instruction of the University and its staff in relation to the commencement of any assessment / examination during my programme of study in Universiti Tunku Abdul Rahman.

I hereby declare that my submission for all assessment / examination during my programme of study in the University shall be based on my original work, not plagiarized from any source(s) except for citations and quotations which have been duly acknowledged. I am fully aware that students who are suspected of violating this pledge are liable to be referred to the Examination Disciplinary Committee of the University.



Signature :

Name : Cha Hong Yip

ID No. : 1804295

Date : 9/9/2022

I, ___Cheah Ho Ching___ (Name), Student ID No. _1801494_ hereby solemnly and sincerely declare and confirm that I have read, understood, and shall abide and comply with all laws, rules, regulations, guidelines and lawful instruction of the University and its staff in relation to the commencement of any assessment / examination during my programme of study in Universiti Tunku Abdul Rahman.

I hereby declare that my submission for all assessment / examination during my programme of study in the University shall be based on my original work, not plagiarized from any source(s) except for citations and quotations which have been duly acknowledged. I am fully aware that students who are suspected of violating this pledge are liable to be referred to the Examination Disciplinary Committee of the University.

Cheah

Signature :


Name : Cheah Ho Ching

ID No. : 1801494

Date : 9/9/2022

I, ___Kelvin Chang Yik Yang___ (Name), Student ID No. _1801255_ hereby solemnly and sincerely declare and confirm that I have read, understood, and shall abide and comply with all laws, rules, regulations, guidelines and lawful instruction of the University and its staff in relation to the commencement of any assessment / examination during my programme of study in Universiti Tunku Abdul Rahman.

I hereby declare that my submission for all assessment / examination during my programme of study in the University shall be based on my original work, not plagiarized from any source(s) except for citations and quotations which have been duly acknowledged. I am fully aware that students who are suspected of violating this pledge are liable to be referred to the Examination Disciplinary Committee of the University.

Signature : 

Name : Kelvin Chang Yik Yang

ID No. : 1801255

Date : 9/9/2022

Project Overview

Game Overview (Written by Cha Hong Yip)

For our game, it is a 2D side-scroller Metroidvania style game. We mainly took inspiration from the gameplay of Hollow Knight, another Metroidvania game. The main gameplay we had in mind is to imitate some of the gameplay of dodging enemy attacks and the control the player has over the game character.

To increase gameplay elements that are available to players, we also took inspiration from Hollow Knight's obstacle course gameplay and implemented obstacle courses in our game. The obstacle course also serves as a tutorial to the player which teaches the basic movement controls to the player.

For our game world setting, it is set in the medieval western era of knights and dragons. For the backstory of our game, it is about a hero that embarked on a journey to infiltrate the monster's castle to defeat the monster's leader that is reigning over the land of the hero. Over the stages in the game, it is about the journey that the hero took to infiltrate and how he defeated the monsters.

In order of the stages,

1. Stage 1 is an obstacle course, which shows the defenses around the castle and how the hero avoided the traps set out by the monsters. Once the hero got into the castle, the hero took off his infiltration gear and took on weapons that are available on the castle to fight.
2. Stage 2 is the gatekeeper of the castle. Once the gatekeeper is defeated, the hero gets stronger from this fight to continue finding the monster leader.
3. Stage 3 is the ambush of monster minions sent by the monster's leader to stop the hero. The hero must survive the ambush before the hero can meet the leader for the final battle.
4. Stage 4 is the room where the leader of the monsters is located. The hero fought the leader and won, which saved the hero's homeland from the reign of monsters.

Game Premise (Written by Cheah Ho Ching)

You Play as an adventurer in the medieval era, who infiltrates the castle to defeat the monsters and its boss to save the world. But before you can fight the monster boss, you will need to get past the obstacle to reach the top of the castle and defeat its minions. You will be able to learn new skills and upgrade your characters along the way as a mage or warrior.

Prominent game features (Written by Kelvin Chang Yik Yang)

1. Stage 1 of the game is an obstacle course; the task of the player is to avoid the obstacles and reach the endpoint.
2. After the player has successfully passed Stage 1 of the game, the player is required to select a character of either warrior or mage to continue the game.
3. Warrior is a melee-attack character with a higher hit point and lower damage, while the mage is a ranged-attack character with higher damage and lower hit point.
4. Both warrior and mage have their own special ability. Warrior is able to activate a shield that can block the damage from the enemies. While the mage can activate the invisibility skill which allows the character to be invulnerable to enemy damage for 10 seconds.
5. In Stage 2, the player must fight the dragon boss. The attack damage of the dragon will be doubled once its hit point drops below 40%.
6. After the dragon boss in Stage 2 has died, a buff potion will drop, and the character gets the power up. Both warrior and mage have two different power-ups, the player may select one out of two power-ups for the character.
7. In Stage 3, the player must kill the minions spawned and survive the attacks from the minions within the given time.
8. In the final stage, the player must fight the final boss, a phoenix which has a higher hit point, smarter behaviour, and the ability to change the world of the battlefield and give random de-buff to the player when its hit point drops below 50%. There is a healing zone for the player to heal themselves when entering the zone.

Team Designation

Member	Role
Lu Jason	The development of the objects of ninja, warrior, and mage. He also creates the animation of each of the game object. He also develops the scene of Start scene, Stage 1 scene, and Select Character scene. He also develops the pause menu for each scene.
Kelvin Chang Yik Yang	Responsible in enemies sprite searching. Developed stage 2 and dragon boss AI. Designed the buffs of warrior and mage and developed the power-up scene.
Cheah Ho Ching	Responsible for developing background sprite. Developed stage 3 with countdown timer, timer pickup and enemies with AI.
Cha Hong Yip	Responsible for developing mage character of the game. Developed stage 4 and Phoenix boss AI. Developed gimmick for stage 4.

Game Engine Details

Game Engine Version and Summary (Written by Lu Jason)

Unity is one of the popular game engines and it is launched in 2004. Unity is available to develop the game in various of the platforms, such as mobile, desktop, iOS, Android and so on. Besides that, this game engine is widely used in the development of mobile game of iOS and Android platform. By using Unity, we can develop the game in 3D, 2D, and interactive simulations as well. In addition, Unity also been applied in different kinds of professional industries, such as Engineering, Construction, Film etc. The version of Unity in this project is Unity 2021.3.4f1. This version was released on 1 June 2022.

In this version of Unity, the programming language used to develop the script is C#. Besides that, other group of classes and APIs can use together with C# as well. C# is widely used in Unity due to it is friendly to the beginner in programming language. This is considered as one of the reasons why Unity is popular for the beginner in game development.

Beyond than that, in this version of Unity it also contains 5 sections which are scene view, game view, hierarchy, project, and inspector. In scene view, users can design their game and levels. Besides, the game objects and design elements can create and apply in the scene view. In game view, user can view their designation outputs. In order to see the results, we must have a camera on the scene. For the hierarchy, every game objects created in the scene will display in the hierarchy section. To display the object in the scene, the object must be registered. For the project section, it shows the elements inside the folder of asset. It includes the elements such as folders, audio, video, game objects, and these elements are allowed to access. For the inspector section, this is a panel in the project. Inside the panel, it contains the properties and attribute of the game object.

Lastly, the new feature added in this version is the vulkan contains the gaze foveated rendering (GFR). Besides, this version also made some improvements such as the AssetBundle extension has been added to the list of StreamingAssets' gradle compression exceptions, implemented the code snippet for

Physics.ContactModifyEvent, all users may now check out of analytics inside the editor, and the settings will synchronize with the backend of the Unity.

Game Engine Features (Written by Cha Hong Yip)

The game engine used is Unity. Unity is a cross-platform game engine, which means it can develop games for multiple platforms which include desktop, mobile, and console. Unity can be used to create 3D and 2D games.

Licensing

Depending on your project or position, the Unity licensing fee differs. As we are just a group of students and not using Unity for commercial work, Unity can be used free of charge, therefore suits our student project financially.

Animation tools

Unity engine provides an animation tool which allows developers to create animation for game objects. Animation tools in the Unity engine can be accessed from programming scripts. The animation tools also include an animator which can control the state of a game object, triggering appropriate animations depending on the actions done by the game object.

Asset Store

Unity provides an asset store where asset developers can put up animation, game assets, or audio for use by others either free of charge or for a fee.

Creating and Destroying GameObjects

The Unity engine can create and destroy game objects whenever it is needed. A game object can be created only when a particular scene is loaded and be destroyed when the game has moved on to another scene. This feature helps to save computing resources of the platform it is on.

Accessing Component

Different components can be assigned to each game object from the Inspector window. These components added can be used to affect how the game object functions and control the game object when the game starts. A few basic components used by our projects are the scripting component where the scripts for a game object is placed, Rigidbody 2D which controls the physics of the game object, Audio Source which controls the audio and sounds for the game object.

Events for GameObject

Unity provides events functions for developers. A few of the events that are used by our game is the regular update event, which is the `Update()` function which runs on every frame to calculate monster and player movement and animations. Another event that is commonly used by our game is a physics event, `OnCollisionEnter()`, which triggers whenever 2 game objects collide and something is supposed to happen such as when an enemy collides with a player's attack.

Scripting/ Programming Language (Written by Cheah Ho Ching)

The scripting language used by Unity is C#. C# is a programming language developed by Microsoft for the .Net framework which together with the Visual Studio IDE.

Scripts are used in Unity to create your own custom component for the GameObjects to have more controls over the gameplay feature, such as trigger game events, modify Component properties over time and respond to user inputs.

Each Script in Unity implements the MonoBehaviour class, which enables 2 important functions of scripting with Unity, the Start and Update function. Start function will be called before the gameplay begins, which is useful for initialization of variables in the object while Update will be called every game frame, which is useful for events that need to be handled over time such as movement and user input.

The variables of the Scripts are editable in the Inspector by declaring it public or using SerializeField. The variable can be easily edited using the Inspector feature and even editable when the game is running, allowing developers to easily tweak gameplay settings.

There are a few important built-in classes in Unity. These classes are commonly used to control the gameObject and gameplay events and function using Scripts. These classes are:

1. **GameObject:** finding gameObject, adding and removing components or modifying the variable of GameObject components.
2. **Object:** Base class of all objects in Unity, allow drag and dropping of Object in the inspector such as GameObject, components such as texture or audio and user created custom objects.
3. **Transform:** works with GameObject position, rotation and scale.
4. **Vectors:** works with 2D, 3D, and 4D points, lines and directions.
5. **Quaternion:** works with absolute or relative rotation.
6. **Time:** allows measure and control of time with framerate of the project.
7. **Matf:** contain commonly used math functions such as trigonometric or logarithmic.
8. **Random:** useful for generating random values.

9. Debug: useful for visualizing information. Used for debugging.
10. Gizmo and Handles: displaying shapes and lines in scene view.

If the built-in classes are not enough, scripts created outside Unity can be included in the form of plug-in. There are two types of plug-in, managed plug-in to work with .Net assemblies, and native plug-in, to work with native code libraries such as operating system calls and third-party code.

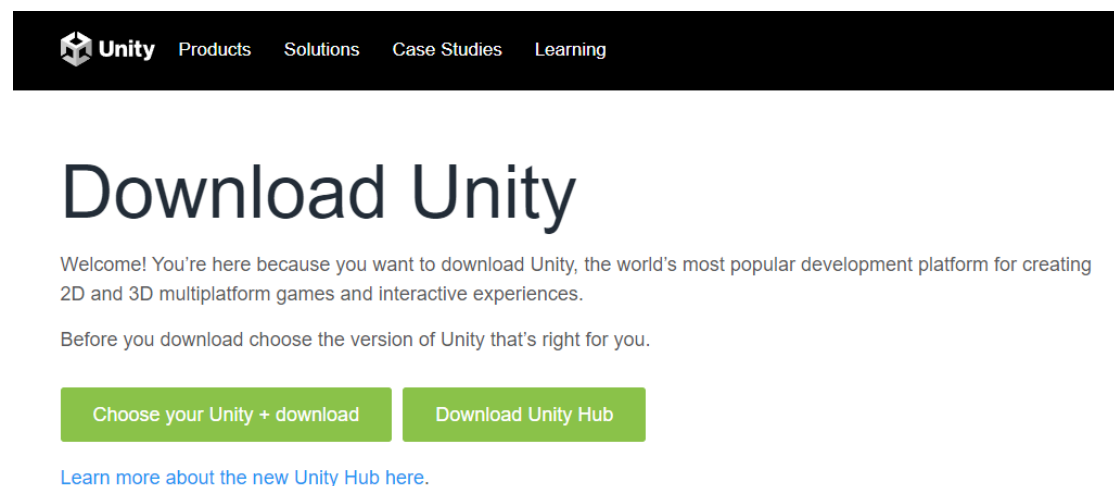
Game Engine Technical Issues and SetUps (Written by Lu Jason)

There are some exist and detected issues in the version of Unity 2021.3.4f1. The issues are listed in below.

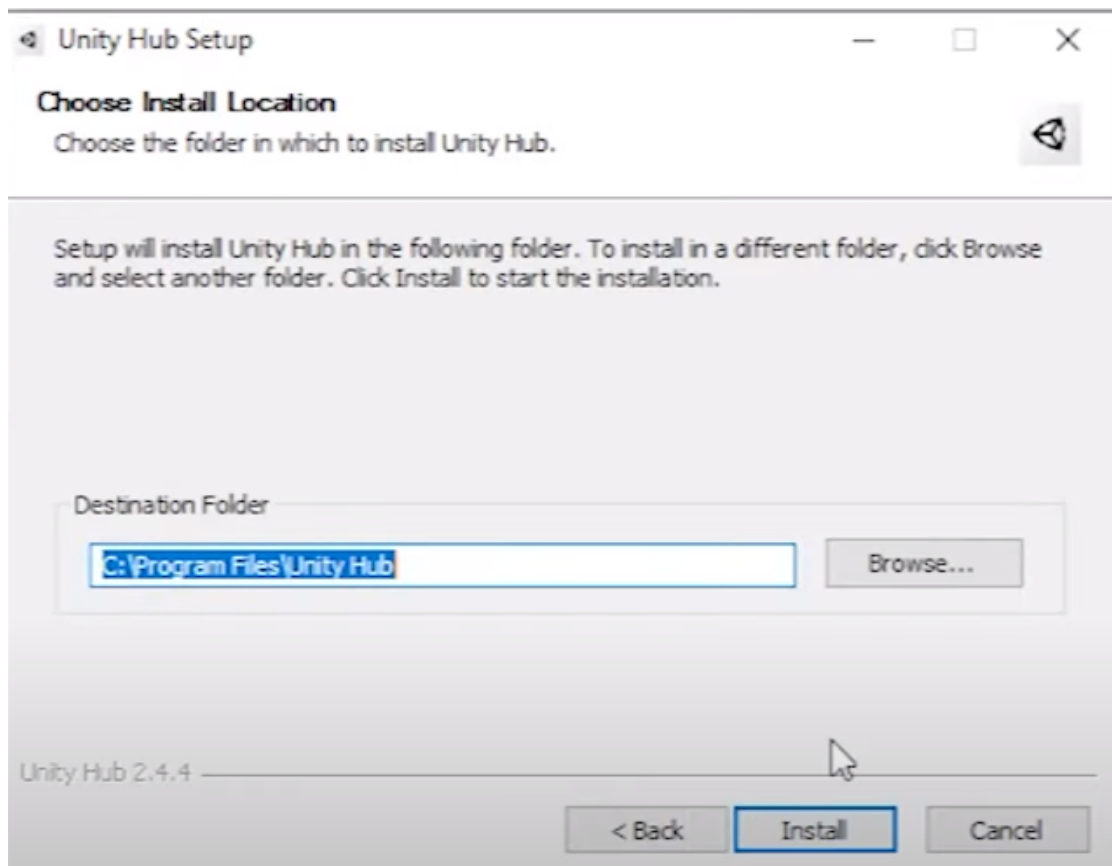
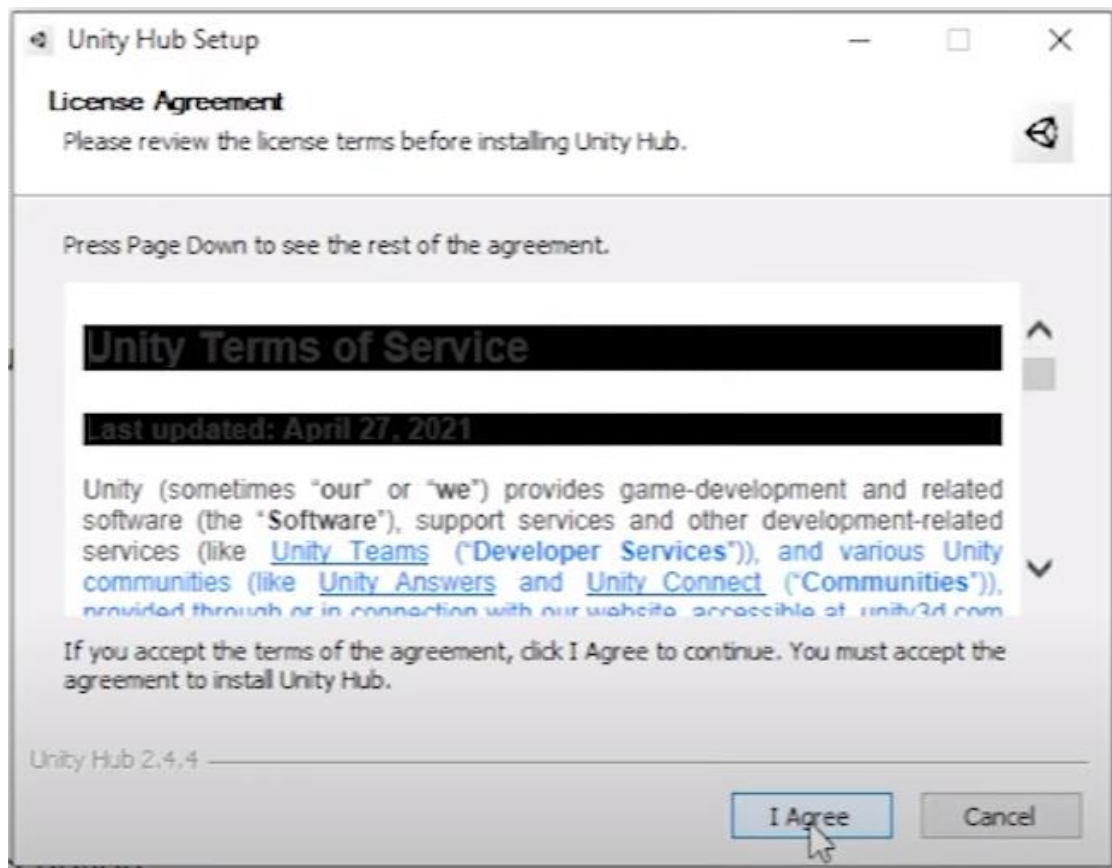
- developer failed to search for the Java when the developer is developing a mobile application for Android platform
- when developing a project which is empty, an empty folder of streamingAssets would create as well.
- For the UAV texture, which is randomly write, it would cause the GPU hang when filtering Scene Hierarchy items.
- When pulling the latest work from GitHub, the sprite would automatically be missing, hence we need to re-link the missing sprite again.

To use this version of Unity as a game engine for our project, there are some steps required to follow. The required steps are listed as below

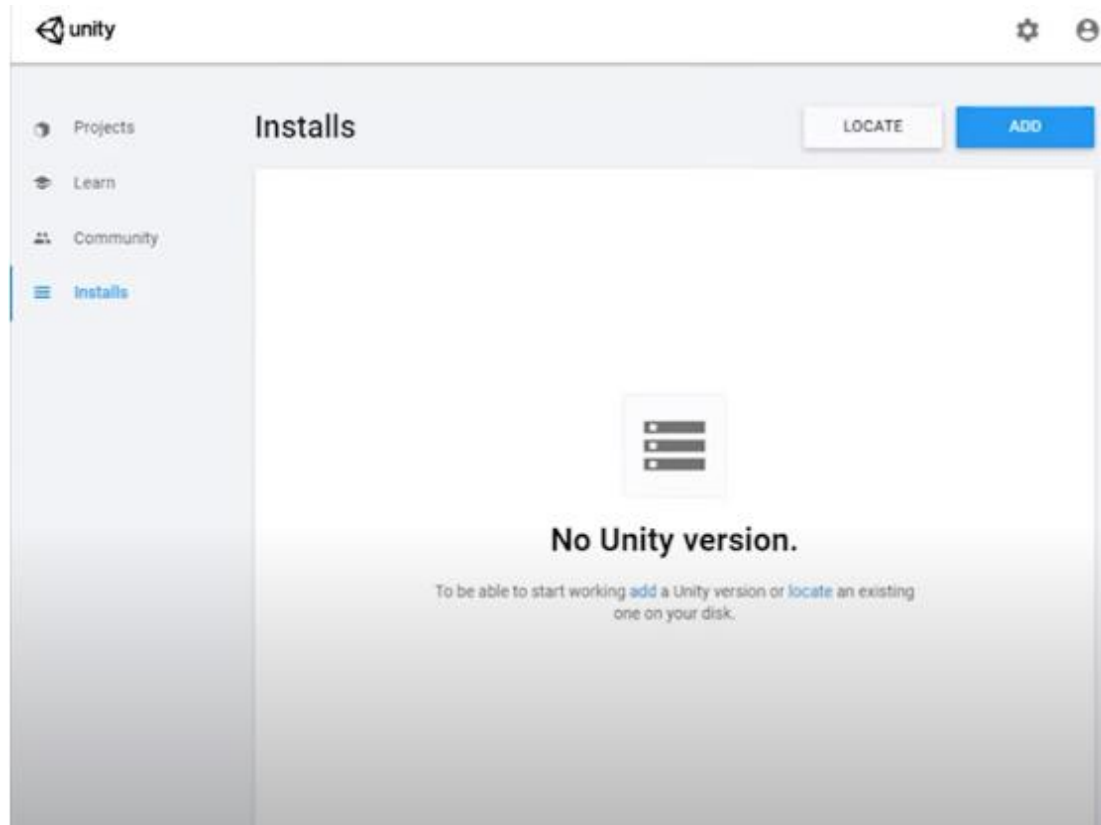
Firstly, download the Unity hub from this link [Download - Unity \(unity3d.com\)](https://unity3d.com).

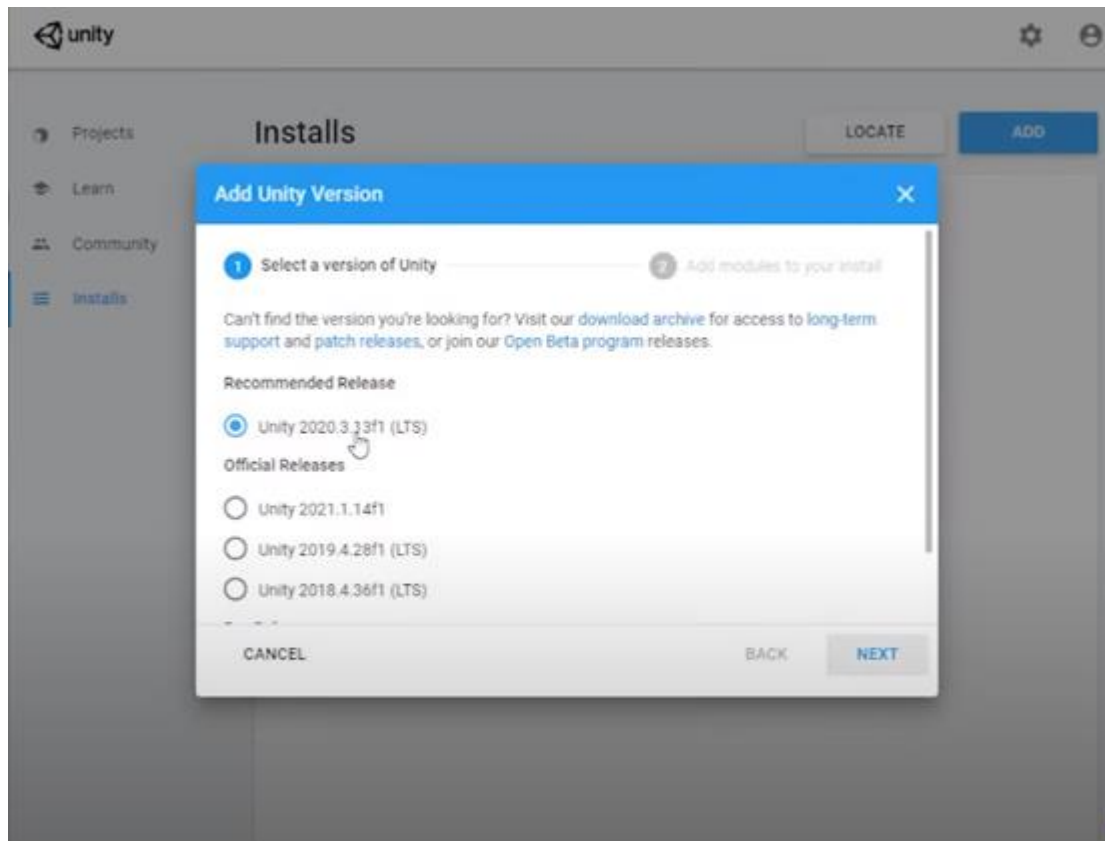


Then complete the installation of Unity hub Setup. By clicking on “I Agree” button for License Agreement and choose the location of the folder where you prefer to place it.

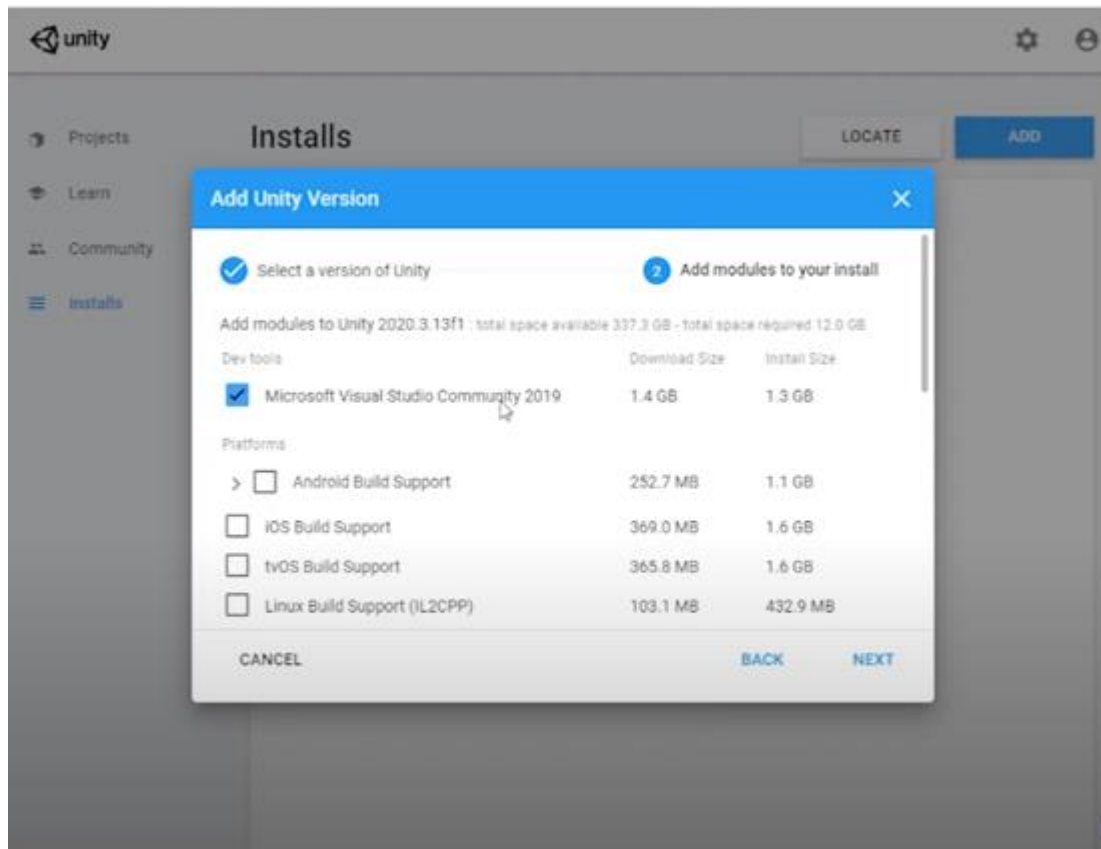


Then we can add the version of Unity 2021.3.4f1 from the Unity hub, to do this we can click on the “Add” button.

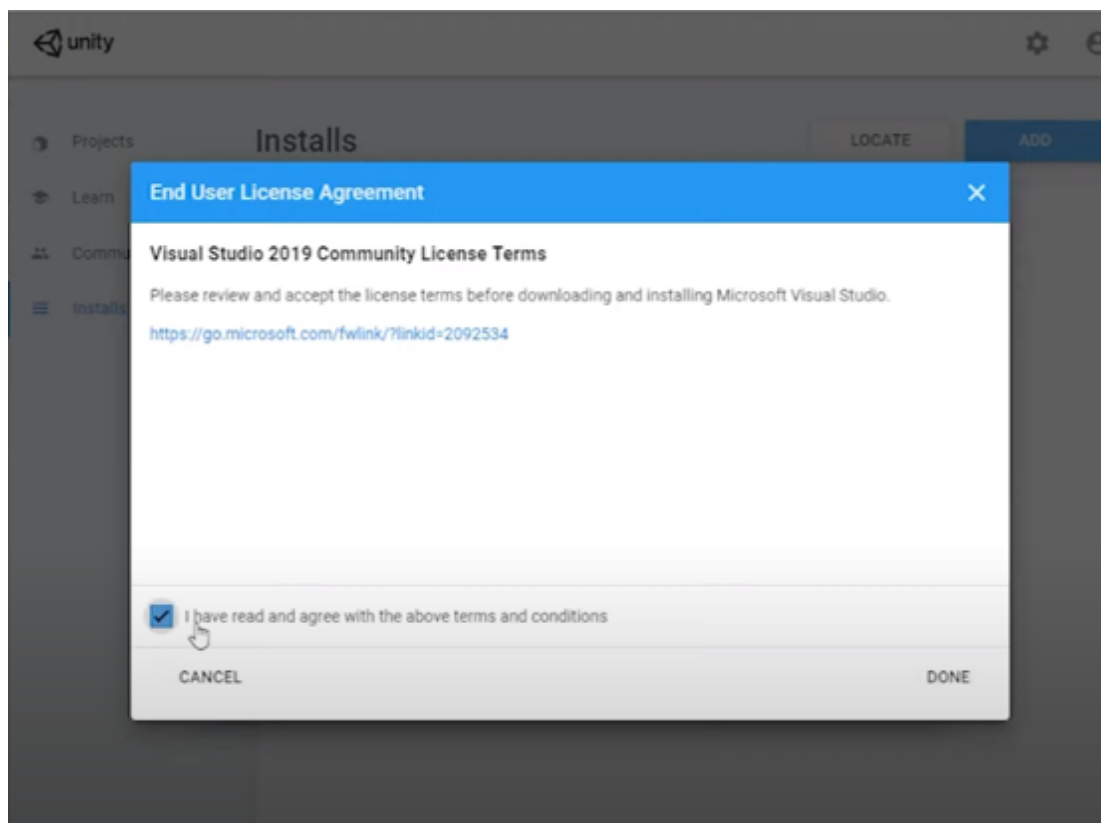




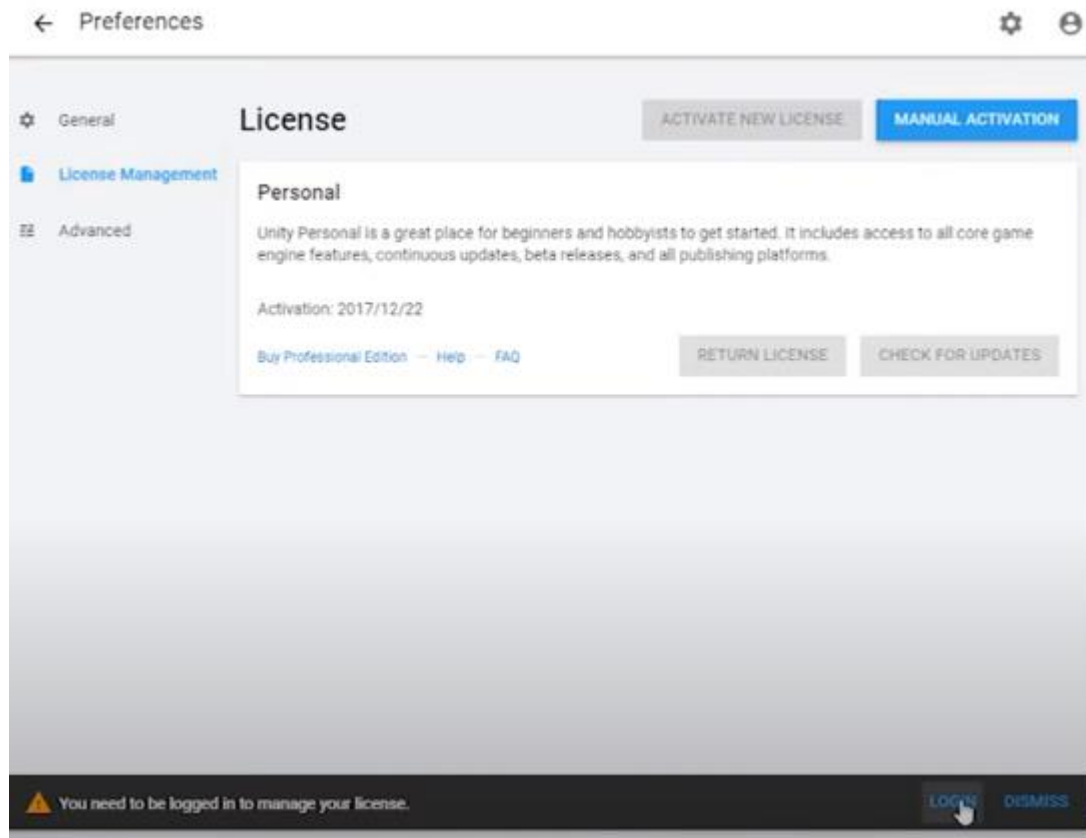
Since all of us have the Visual Studio Code as our code edit tools, so we would cancel the installation of Microsoft Visual Studio Community 2019. Then click on the “Next” button to proceed.



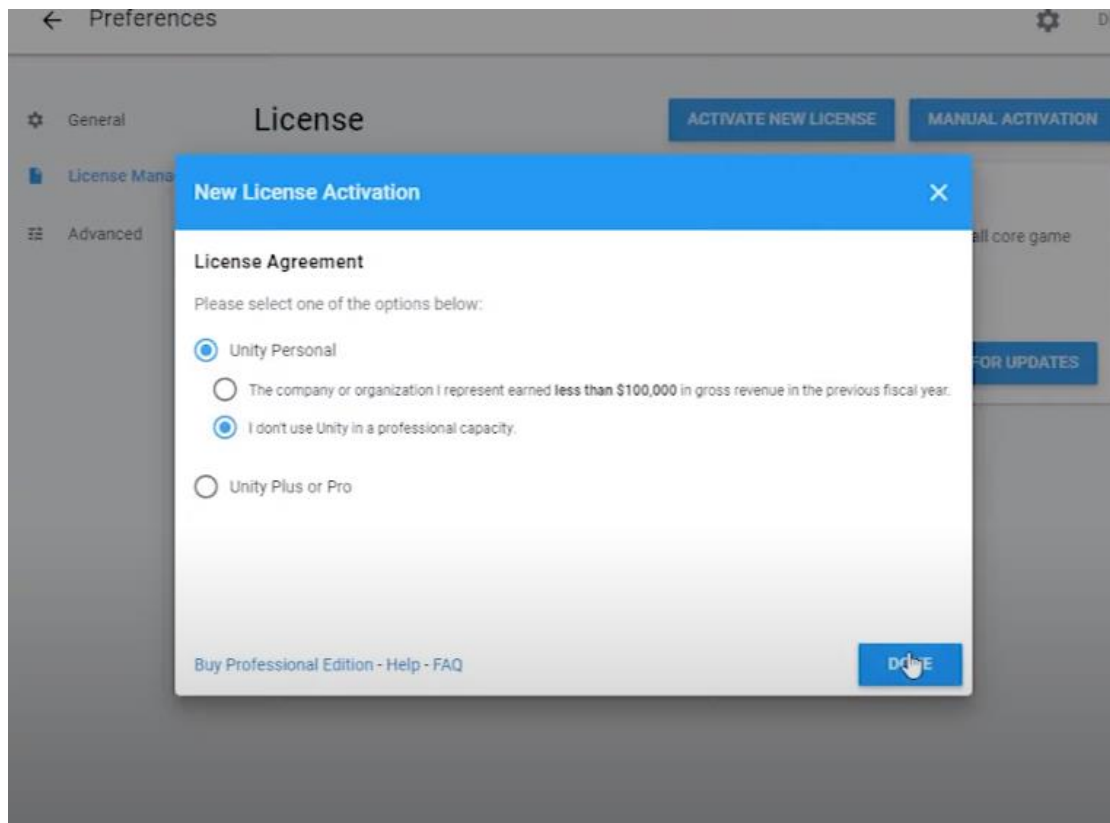
Then click on the “I have read and agree above terms and conditions” button to complete the installation.



Now the version would start to download, once the version is successfully download, we need to get the licenses as well. By clicking on the settings icon on the top left corner, then select the Licenses Management sector, and click on “Login” button on the bottom notification.



Then login to your own Unity account and click on the “ACTIVATE NEW LICENSE” button to get the license. After clicking the button, a new window will pop out, then select Unity for personal and I don’t use unity in a professional capacity.



Then click on the “DONE” button to finish the license settings. Now we can start to create the project in Unity.

Hardware Technical Specifications & Requirements (Written by Kelvin Chang)

Specification and Requirement	Windows	macOS
OS Version	Windows 7 (SP1+), Windows 10, and Windows 11 64-bit versions only.	High Sierra 10.13+ (Intel editor) Big Sur 11.0 (Apple silicon Editor)
CPU	X64 architecture with SSE2 instruction set support	X64 architecture with SSE2 instruction set support (Intel processors) Apple M1 or above (Apple silicon-based processors)
Graphics API	DX10, DX11, and DX12-capable GPUs	Metal-capable Intel and AMD GPUs
Additional Requirements	Drivers that are officially supported by hardware vendors.	Drivers that are officially supported by Apple. (For Intel processor) Requires Rosetta 2 for both Apple silicon-based processor and Intel processor.

Table 2.1 Minimum System Requirements of Unity (Unity, n.d.)

Project Technical Design Overview

Game Technical Specifications (Written by Cha Hong Yip)

Minimum requirements	Windows
Operating system version	Windows 7 (SP1+) and Windows 10, 64-bit versions only
CPU	X64 architecture with SSE2 instruction set support
Graphics API	DX10, DX11, and DX12-capable GPUs
Additional requirements	Hardware vendor officially supported drivers

Information taken from Unity documentation

Project Game Development Setup

Project Organization Overview (Written by Lu Jason)

Inside the Asset folder:

- All of the materials related to animations are store under the folder of Animations.
- For the AstarPathfindingProject folder, it stores the material related to pathfinding.
- For the Background folder, it stores the image of background for all scenes.
- For the BGM folder, it stores all of the BGMs.
- For the Physics Material folder, it stores the physics Material 2D.
- For the Prefabs folder, it stores all of the prefabs material created in this project.
- For the scenes folder, it stores all of the scenes.
- For the script folder, it stores all of the script.
- For the Sprites folder, it stores all of the sprite sheet and image.

In the below following, it is showing the project folder structure.

Animations:

- Jason Animation
- Kelvin Animation
- Phoenix

AstarPathfinding Project

Background

BGM

Physics Material

Plugins

Prefabs

- Jason Prefabs

Scenes

Scripts

Sprites

- BackGround
- Checkpoint
- Enemy
- Ground
- Health
- Item
- Kelvin Sprite
- Knight
- Mage
- Normal User
- Portal
- Rogue
- Shield
- Sprites Profile
- Warrior
- Traps

TextMesh Pro

Game Flow/ Scenes Overview (Written by Lu Jason)

Firstly, the scene display is the start scene of the project, in this scene user can click on the “Start” button to start the game.



Figure of start scene

Then, the users will redirect to the stage 1 scene, in this scene user requires to pass through the obstacles and traps in this scene.



Figure of scene of stage 1

In the end of this scene, user will touch the flag and redirect to the scene to select the character. In the scene of select character, user requires to pick the character they prefer. The characters are warrior and mage, for warrior it is close range attack and have a longer hp which is 150, while mage is long range attack, and it has a regular hp which is 100.

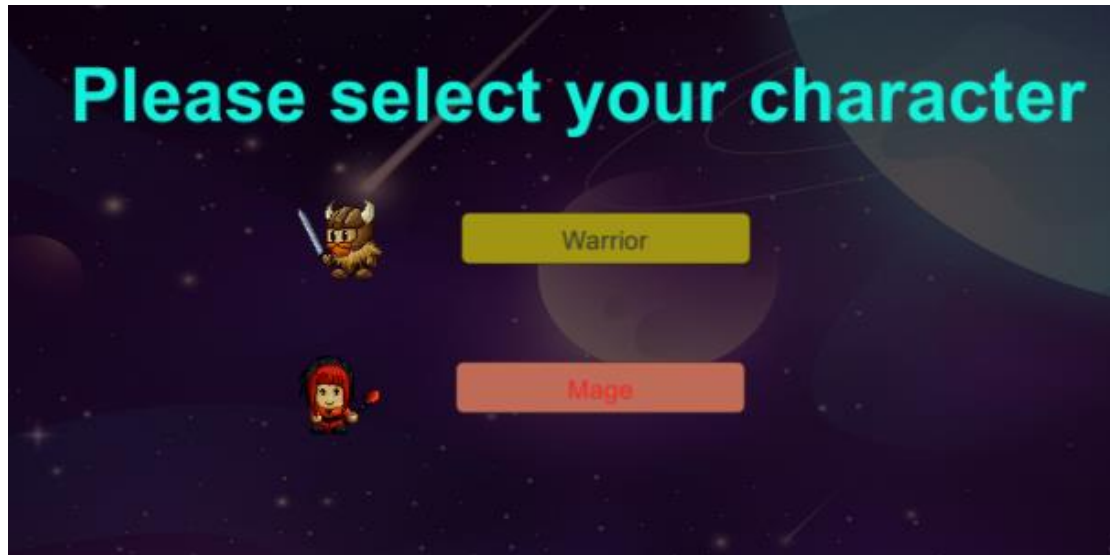


Figure of scene of select character

After the user selected their character, they will redirect to the scene of fighting against the enemy boss 1. The enemy boss 1 is a dragon, it will spread the fire to attack the user. In this scene, user requires to kill the dragon.



Figure of scene of fighting against the enemy boss 1

Once the enemy boss 1 is die, then the user will redirect to a scene which allow to pick the extra power/ buff for the character they selected. In this scene, it contains 2 kinds of extra power/ buff.

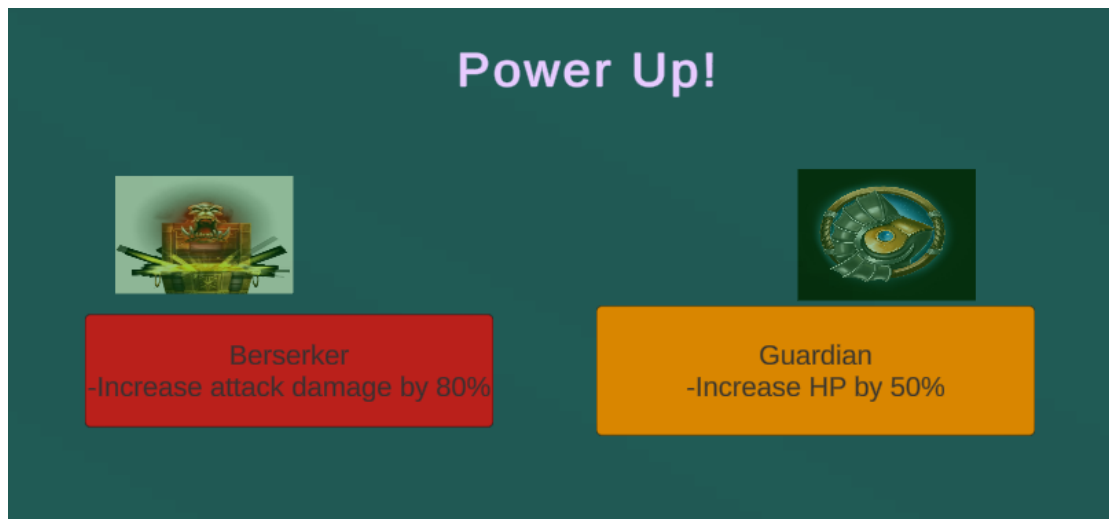


Figure of extra power/ buff scene for warrior character

Once the user selected the extra powers/ buff, then user will be redirect to the scene to fight against multiple of eagles. The purpose of this scene is requiring the user to survive in a specific time. In this scene, the countdown timer will display to remind the user how many times left. If the user dies in this scene, they will respawn again, and the time will increase as a punishment.

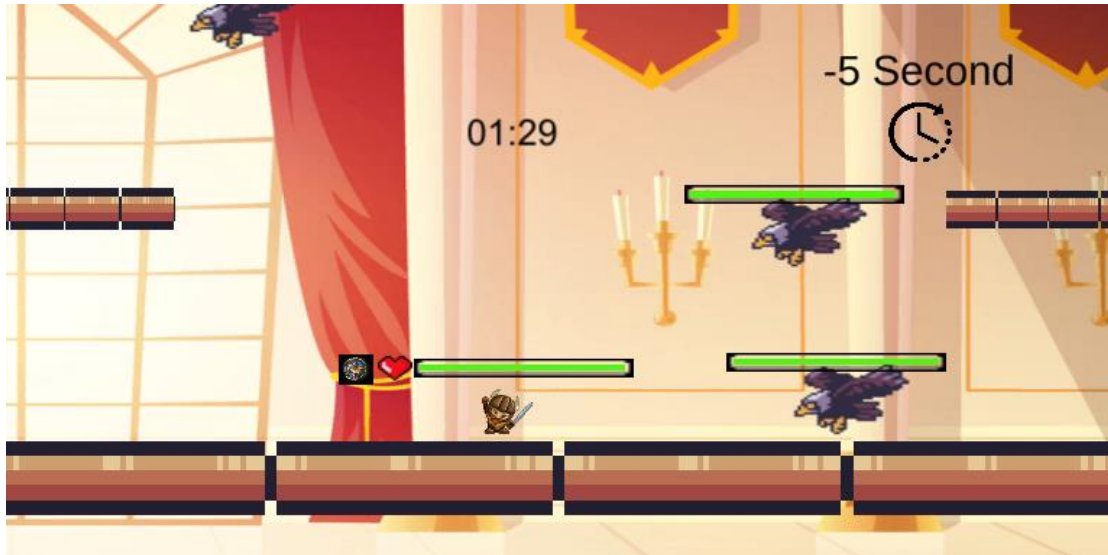


Figure of scene of user to fight against eagles and survive through the specific time

Once the user survive until the time is up, then it considers as success and it will redirect the user to the final scene. In the final scene, user need to fight against the final boss enemy. This final boss is a phoenix, and it contains 2 types of attack mode. One of the attack modes is fire bullet, this fire bullet will deploy once the phoenix found the location of the user. Another attack mode is random deployment of fireball, this random fireball will deploy by phoenix randomly. Once the user successfully killed the phoenix in this scene then the whole game is over, and a canvas of game end will display as well.

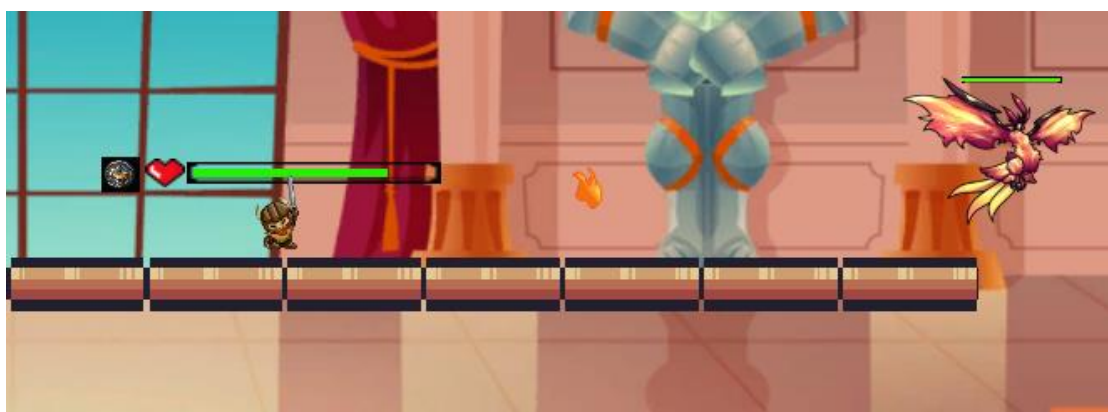


Figure of scene of fighting against the phoenix

In this project, we arranged the scenes in the build settings in ascending mode. The order of the scenes is listed as below.

1. Start scene
2. Scene of stage 1
3. Scene of select character
4. Scene of enemy boss 1
5. Scene of extra power/ buff for Warrior character
6. Scene of extra power/ buff for Mage character
7. Scene of fight against multiple of eagle
8. Scene of fight against final boss enemy

In the build settings, the index of the scene is starting from 0. For example, the start scene is the first scene, so it has the index of 0, and the stage 1 scene is second, so it has the index of 1. For the index of the rest of the scenes, it is increasing accordingly.

Programming Conventions and Designs

Overall Script Setup

In this project, the total of scripts is 48. Each of the scripts are using C# as programming language to develop. In the start scene, the Start button apply the function of StartGame() in the script of StartMenu. In the stage 1 scene, the player object applies the scripts of Player Movement, Player Life, Grapple Hook, Ladder, and Player Teleport scripts. One of the traps in this scene, which is jigsaw apply the script of moving platform, so that the jigsaw can moving back and forward in the game. For both portals apply the script of teleporter, so that user can transform themselves by using the portal. Besides in this scene the moving platform which allow the user to pass through the spikes, it is applying the script of moving platform and sticky platform. Furthermore, the flag inside this scene applies the script of Finish, when user touch this flag it would redirect the user to the scene of select character.

In the scene of select character, it applies the functions of SelectMage and SelectWarrior in the script of StartMenu. Besides, in this scene the warrior object applies the script of Player Movement, Melee Attack, Shield and Warrior Life. While for the mage object, it applies the script of Player Movement, Attack, Power, and Mage Life.

In the stage 2 scene, the scripts of Dragon, DragonBreath and DragonWalk are applied on the dragon object. The item dropped after the death of dragon applies the BuffPotion script which loads the power-up scene. In the power-up scene, the PowerUp script is applied to allow user to choose a power-up for the character chosen.

In the stage 3 scene, the scripts responsible for the Countdown timer is Timer.cs and TimerEndUI.cs, which control the countdown of the timer and the game over screen when the countdown reach 0. For the Eagle enemies, there are Enemy.cs, which control the respawn and take damage, EnemyPatrol.cs which control the movement of Eagle that patrol between 2 point, Chasing.cs which control Eagle to chase toward the player, EnemyPatrolAI.cs which control patrolling Eagle to attack player on sight.

In stage 4 scene, the Pheonix have applied Seeker.cs, EnemyPathFinding.cs, PheonixAI, Enemy.cs, and SceneController.cs, to create the AI for the Pheonix. The bullets shot by Pheonix have bullet.cs applied which tracks the player's location before

being shot out. The steel ball attack spawned by the Pheonix have RandomFireBall.cs applied. The healing zone have HealingZone.cs applied which heal up player's health when entered the surrounding area.

Coding Conventions and Organization

The naming conventions of classes and methods are applying the PascalCase, while for the naming conventions of variables is using the camelCase. For the layout in the script, we would declare one line for each of the variables, statement, and declaration as well. For the comment in the script, we would write the comment with the double delimiter // at beginning.


Naming Conventions (Written by Lu Jason)




Naming conventions are one of the important parts when we are developing our code. Besides, naming conventions are considered as a standard to name the classes, methods, and variables as well. There are three types of naming methods which are Camel Case (camelCase), Pascal Case (PascalCase), and Underscore Prefix (_underScore). For Camel Case, the first letter of the word is always in small letter then each word after the first word will starts with a capital letter. For Pascal Case, the first letter of each of the words is start with capital letter. For the Underscore Prefix, the word after the underscore (_) will apply the Camel Case approach.

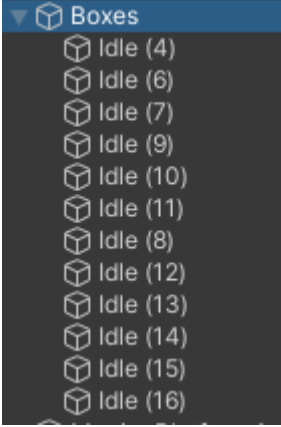

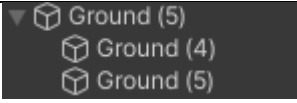
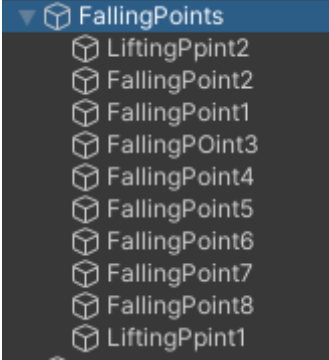
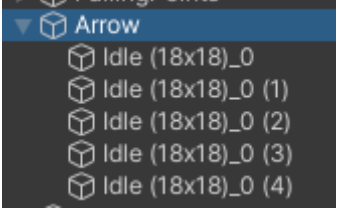

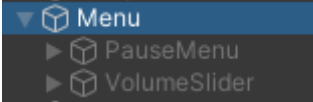
In this project, our code will name the variable using the Camel Case terminology, while the methods and classes will use Pascal Case terminology.

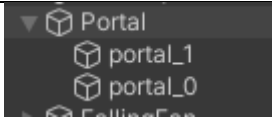

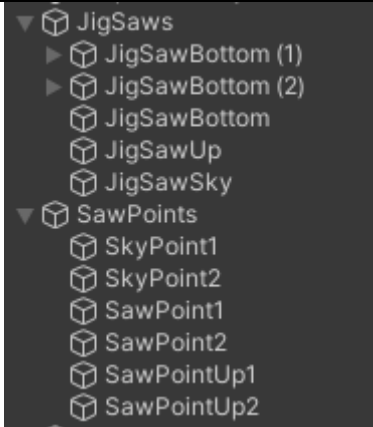
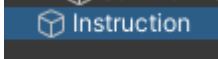
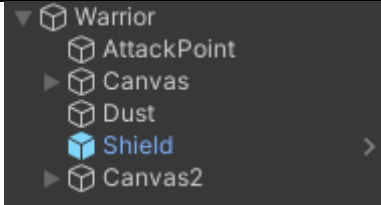
Feature Implementation Summary


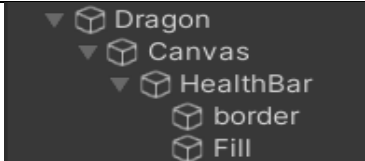

Game Object Design Setup (Written by All)

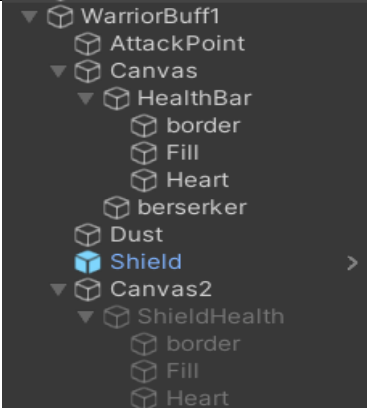

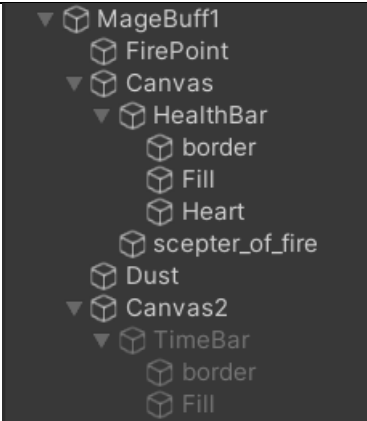
Stage	Name	Image of the game object	Configurations
1	Normal user		<ul style="list-style-type: none"> The canvas is for the of the health bar the dust is the particles generate when user jump

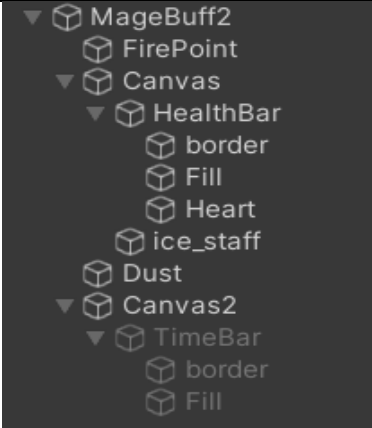
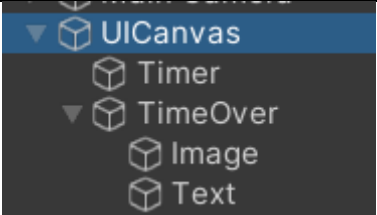
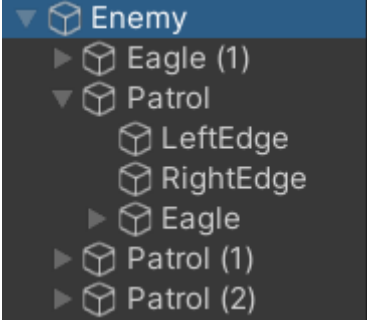

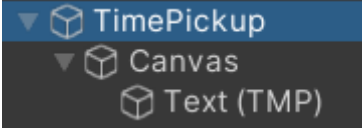
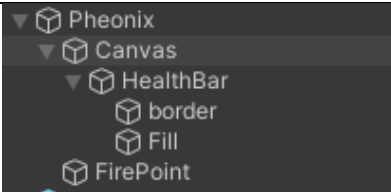
			<ul style="list-style-type: none"> the ground check is to check whether the user is on the ground or not the crosshair is to help the user to aim on the block
1	Spiks		All of the spikes are store in the object of Spikes
1	backGrou nd		The object of background image
1	Skybox		All of the boxes in the sky are store under the object of Skybox.

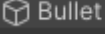




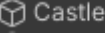

1	Box		All of the boxes on the ground are store under the object of box
1	MovingPlatform1		<p>The moving platform , waypoint 1, and waypoint 2 are store under the MovingPlatform1.</p> <p>When the platform touches the waypoint 1, it will return and moving to waypoint 2</p>
1	Ground		All of the grounds are store under Ground
1	FallingPoints		These are all the points of falling fan.
1	Arrow		The arrow sign which tells the user which way to move on are store under the Arrow.
1	JumpPad		All of the trampolines are store under the JumPad
1	Menu		Canvas of the menu

			<p>Inside this canvas, it consists of PauseMenu and VolumeSlider</p> <p>Pausemenu will shows the button of Resume, Menu, and Options.</p> <p>While user clicks on the Options button, the volume slider will display.</p>
1	Portal		All of the portals are store under the Portal
1	FallingFan		The falling fan is trap, and all of them are store under the FallingFan.
1	JigSaw and SawPoints		<p>The JigSaw store all of the jigsaw in the game, while the SawPoints store all of the start points and end points for the jigsaw to move in between.</p> <p>When the jigsaw touches the start point, it will return to move towards the end point.</p>
1	Instruction		<p>This object is the image of the instruction.</p> <p>It tells the user what is the key to move the player.</p>
2	Warrior		<ul style="list-style-type: none"> • The attack point is the direction of the user currently facing. • The canvas is the health bar of the warrior.

			<ul style="list-style-type: none"> • The dust is the particle generate when warrior jump • For the shield prefabs, it will activate when the user clicks on “TAB” button. This is the power of warrior character • The canvas2 is the hp of shield.
2	Mage		<ul style="list-style-type: none"> • The fire point is the direction of the user currently facing. • The canvas is the health bar of the mage. • The dust is the particle generate when mage jump • When user activate the power of invisible, the canvas 2 will display. It represents the countdown of time.
2	Dragon		<ul style="list-style-type: none"> • The canvas contains the health bar of the dragon.
2	Buff Potion		<ul style="list-style-type: none"> • This is the buff potion that will drop after the dragon dies.

			<ul style="list-style-type: none"> When player collides with the buff potion, the player will be redirected to power-up scene.
2	Warrior Buff 1	 <pre> ▼ WarriorBuff1 AttackPoint ▼ Canvas HealthBar border Fill Heart berserker Dust Shield ▼ Canvas2 ShieldHealth border Fill Heart </pre>	<ul style="list-style-type: none"> The warrior with berserker buff. Increased damage. Berserker buff icon is displayed beside the health bar.
2	Warrior Buff 2	 <pre> ▼ WarriorBuff2 AttackPoint ▼ Canvas HealthBar border Fill Heart guardian Dust Shield ▼ Canvas2 ShieldHealth border Fill Heart </pre>	<ul style="list-style-type: none"> The warrior with guardian buff. Increased maximum health. Guardian buff icon is displayed beside the health bar.
2	Mage Buff 1	 <pre> ▼ MageBuff1 FirePoint ▼ Canvas HealthBar border Fill Heart sceptre_of_fire Dust ▼ Canvas2 TimeBar border Fill </pre>	<ul style="list-style-type: none"> The mage with sceptre of fire buff. Increased fireball damage. Sceptre of fire buff is displayed beside the health bar.

2	Mage Buff 2		<ul style="list-style-type: none"> • The mage with ice staff buff. • Shoots fireball that slows down the speed of enemies by 30% for 2 seconds. • Ice staff buff is displayed beside the health buff.
3	Timer UICanvas		Timer that counts down. The stage will end when the timer is counted to 0. Also include a TimeOver screen indicate the stage is won.
3	Enemy		The enemies of the stage. The eagle that chase the player and eagle that patrol in a constant path. LeftEdge and RightEdge represent the path of the patrol.
3	Terrain		The platform objects of the stage.
3	Time Pickup		The time pickup object that reduce the time remaining when user pick it up. The time pickup will also respawn in random location after a set amount of time.
4	Pheonix		<ul style="list-style-type: none"> • Pheonix that serve as final boss.

			<ul style="list-style-type: none"> • Canvas contains the health bar. • FirePoint is where steel ball attack of the Pheonix will spawn.
4	Bullet	 Bullet	Fireball bullet shot by Phoenix that will damage the player character on collision.
4	Healing Zone	 Healing Zone	An area where when player character enters, the player's health will start to heal.
4	Space	 Space	Background of low gravity negative buff applied to player by Pheonix.
4	Swamp	 Swamp	Background of poison negative buff applied to player by Pheonix.
4	Victory	 victory	Victory Screen when player defeats the Pheonix.
4	Castle	 Castle	Main background of the stage 4.
4	Soil	 Soil	Acts as the ground and wall for the stage.

Player Controls Implementation Overview (Written by Lu Jason)

Key Input	Actions of the player	Flow
ESC	Pause the game first then display the pause menu	<p>User can press on the ESC button to pause the game.</p> <p>When the ESC button pressed, the pause menu will display.</p> <p>The pause menu contains of 3 buttons, resume</p>

		<p>button, menu button, and option button.</p> <p>Resume button will resume the game, menu button will redirect user back to menu scene, option button allow the user to adjust the volume.</p>
A	Move left	User can press on A to move left
D	Move right	User can press on D to move left
LSHIFT	Dash	<p>When user press on LSHIFT, they can dash to the direction they are facing currently.</p> <p><code>rigidbody.velocity</code> is the function to allow the user to move to a location.</p>
Spacebar	Jump	<p>User can press on Spacebar to normal jump, double press on Spacebar to double jump</p> <p>Same as dash, jumping apply the <code>ridigibody.velocity</code> to let the player jump.</p>
TAB	Initiate the power of Shield and Invisible	For the character of Warrior, when the TAB button is pressed, the power of shield will activate

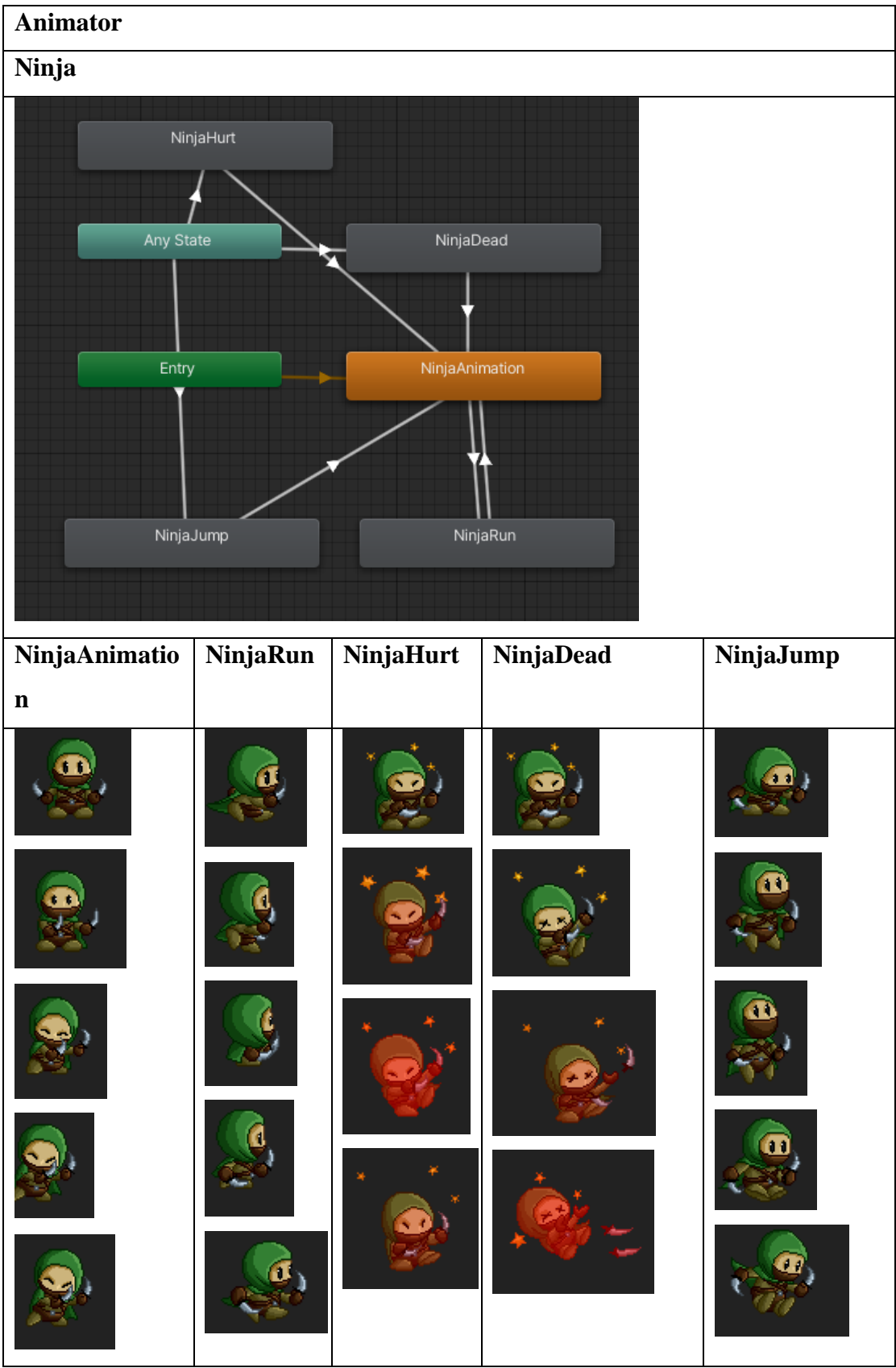
		For the character of Mage, when the TAB button is pressed, the power of become invisible will activate
--	--	--









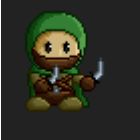







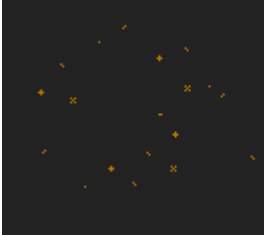
Game Physics Implementation Overview

Jump Physics and Implementation (Written by Lu Jason)

All of the characters in this game can normally jump and double jump. In order to let the player double jump, we will use the Boolean `canDoubleJump` to check whether the double jump is true or false. Initially, the Boolean is set to false. When the user jumps, the bool will change to true. If it is true, the player is allowed to double jump, else player is not allowed to double jump. The height of jump is adjusted by the `jumpStrength` variable in the script. The higher the `jumpStrength`, the higher the player jump.

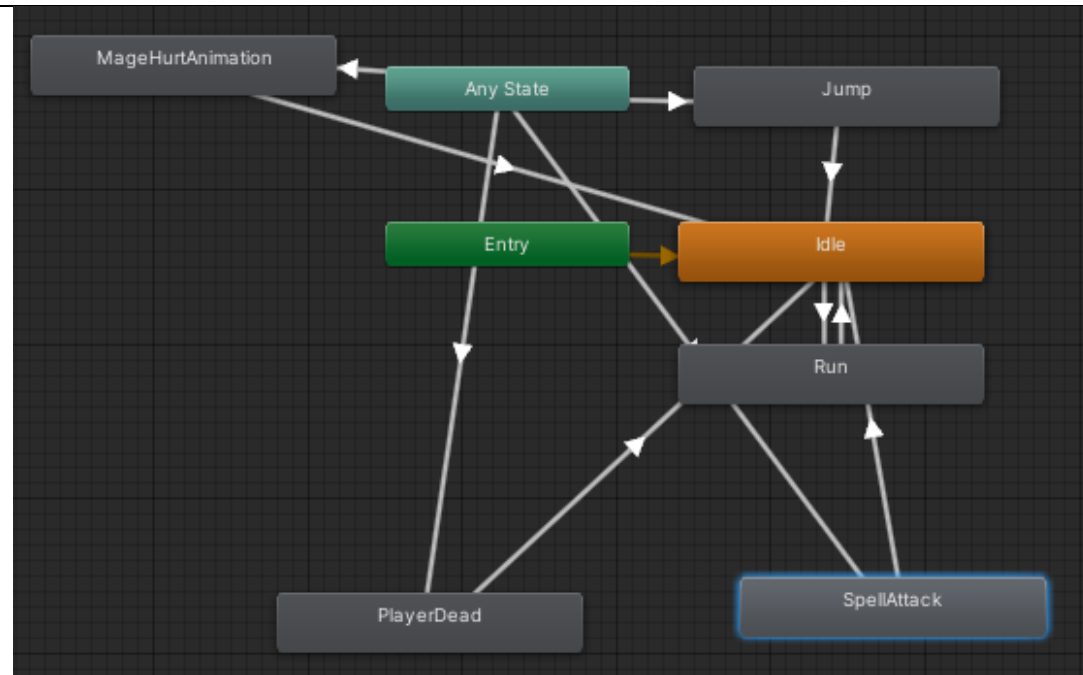
Animation Implementation Overview (Written by All)







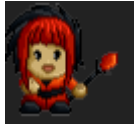


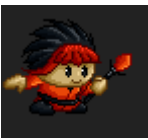





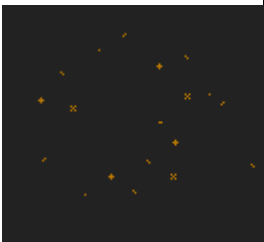




				
				
				
				
				
				

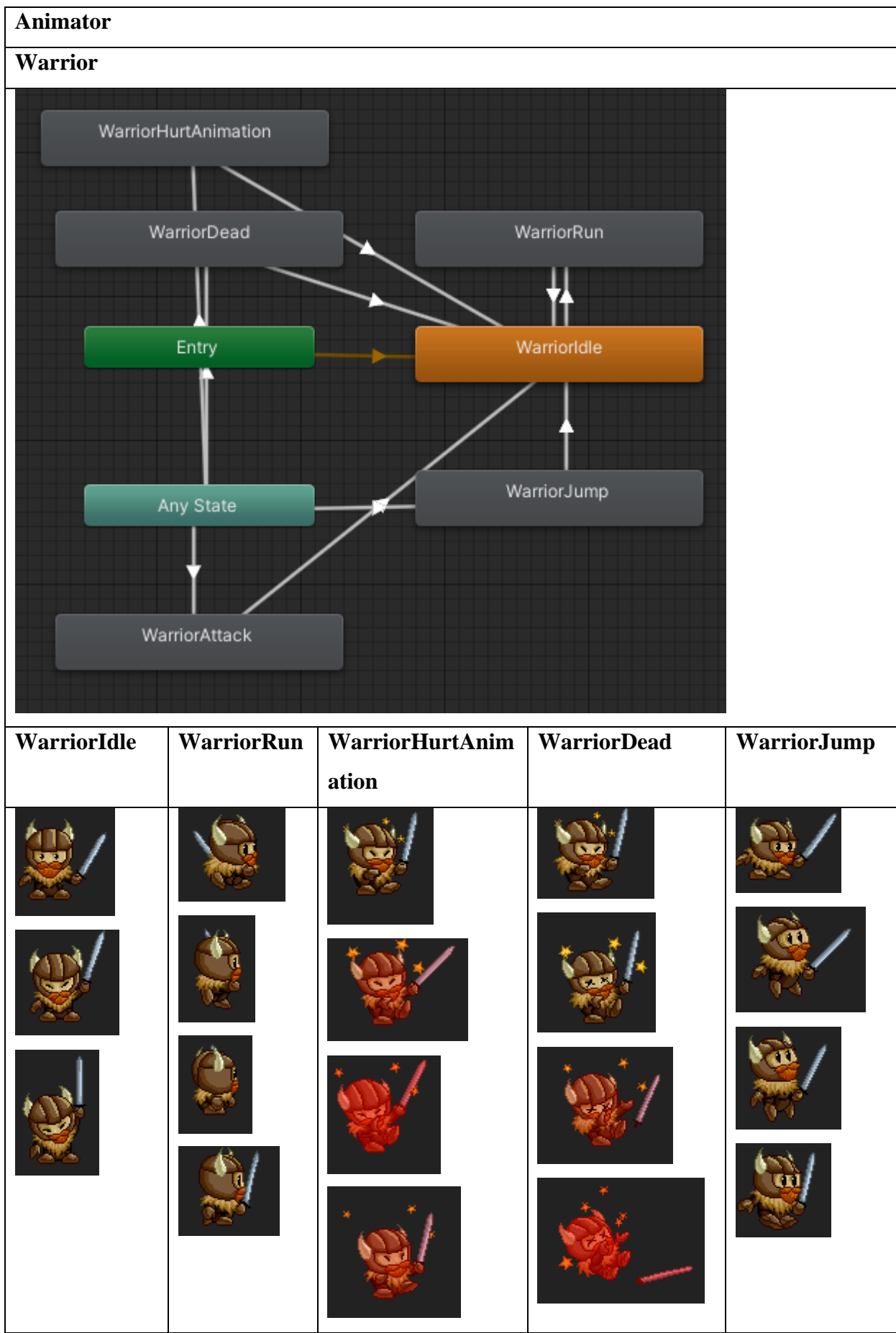
Animator




















Mage



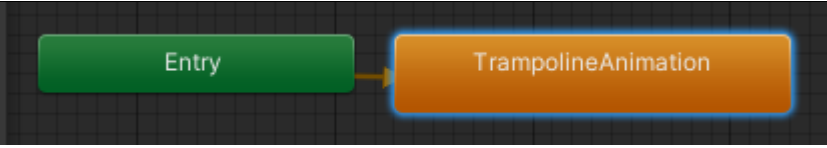
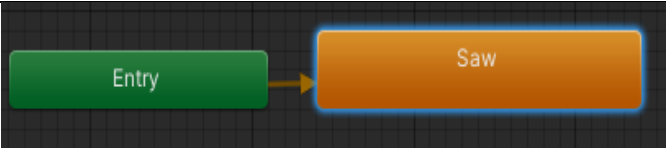


MageIdle	Run	MageHurt Animation	PlayerDead	Jump

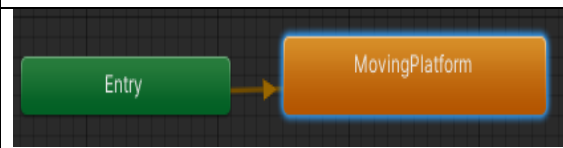


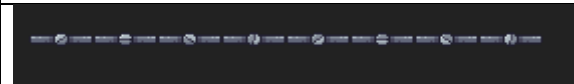
Animator 	Animator 
FallingPlatformAnimation 	PortalAnimation 

<p>Animator</p> 	<p>Animator</p> 
<p>TrampolineAnimation</p> 	<p>Saw</p> 

Animator



MovingPlatform



Animator



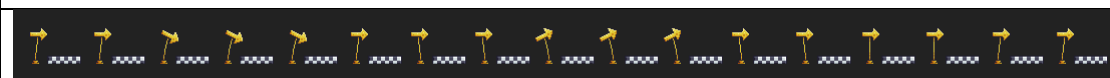
RockTrapAnimation



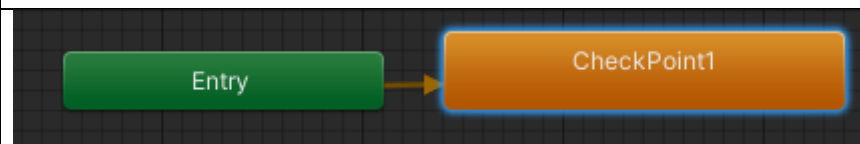
Animator



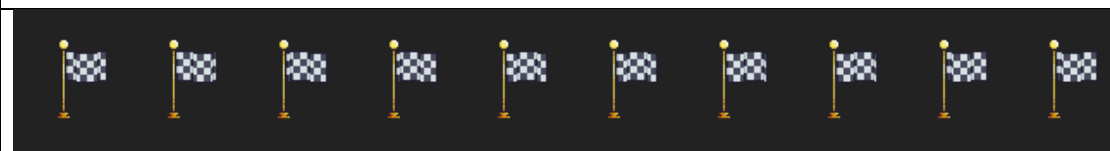
StartAnimation

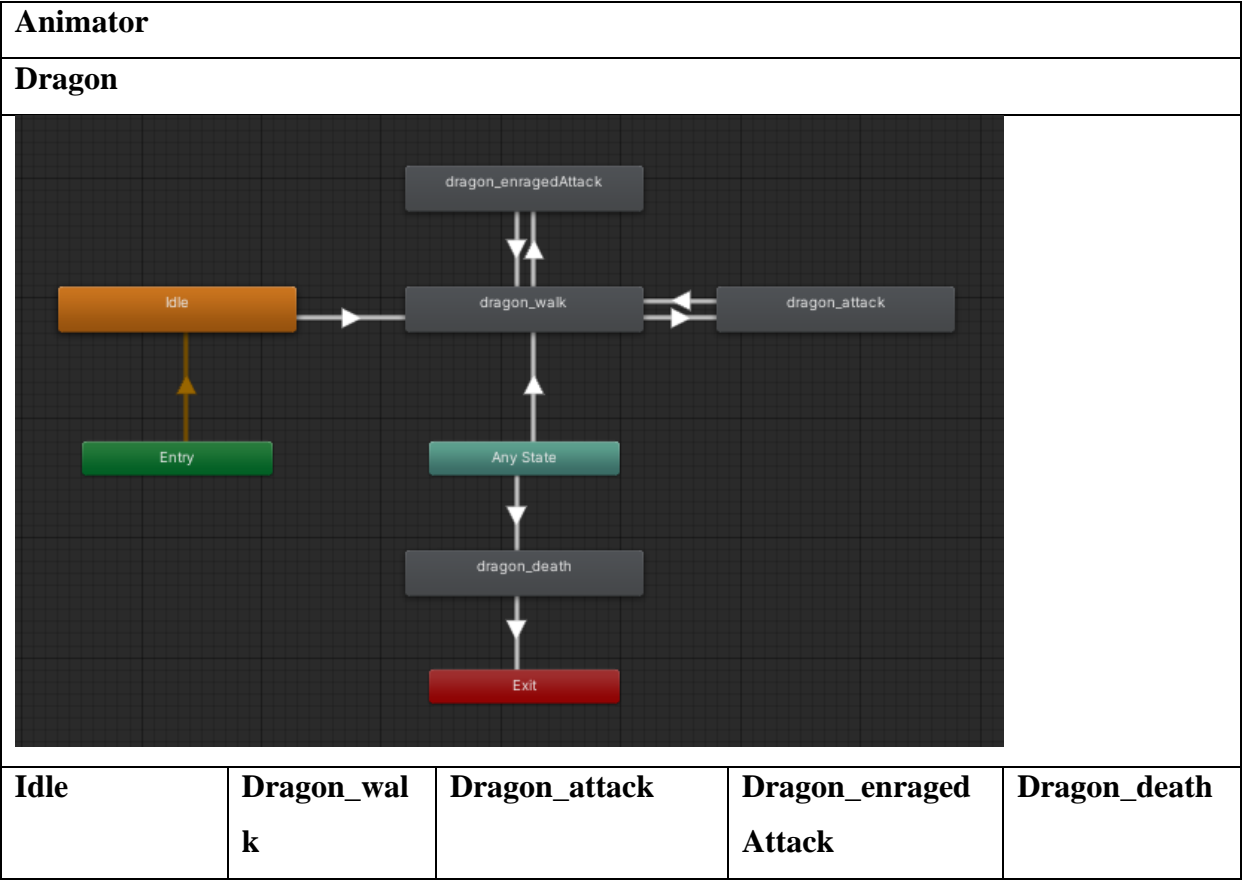
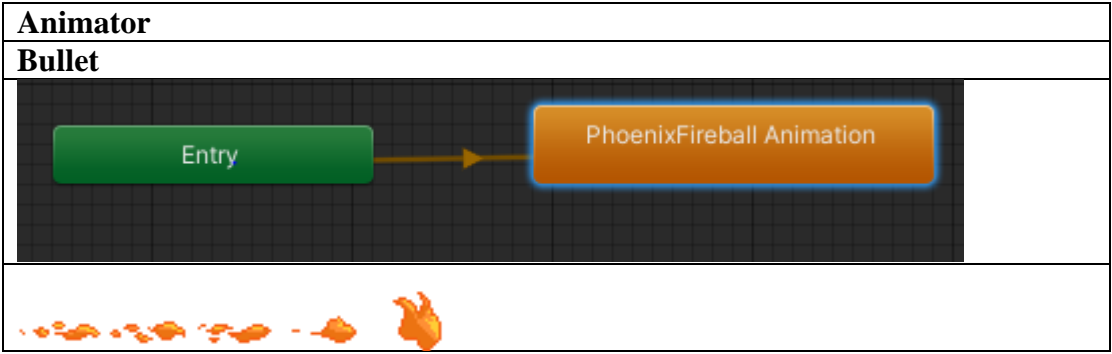


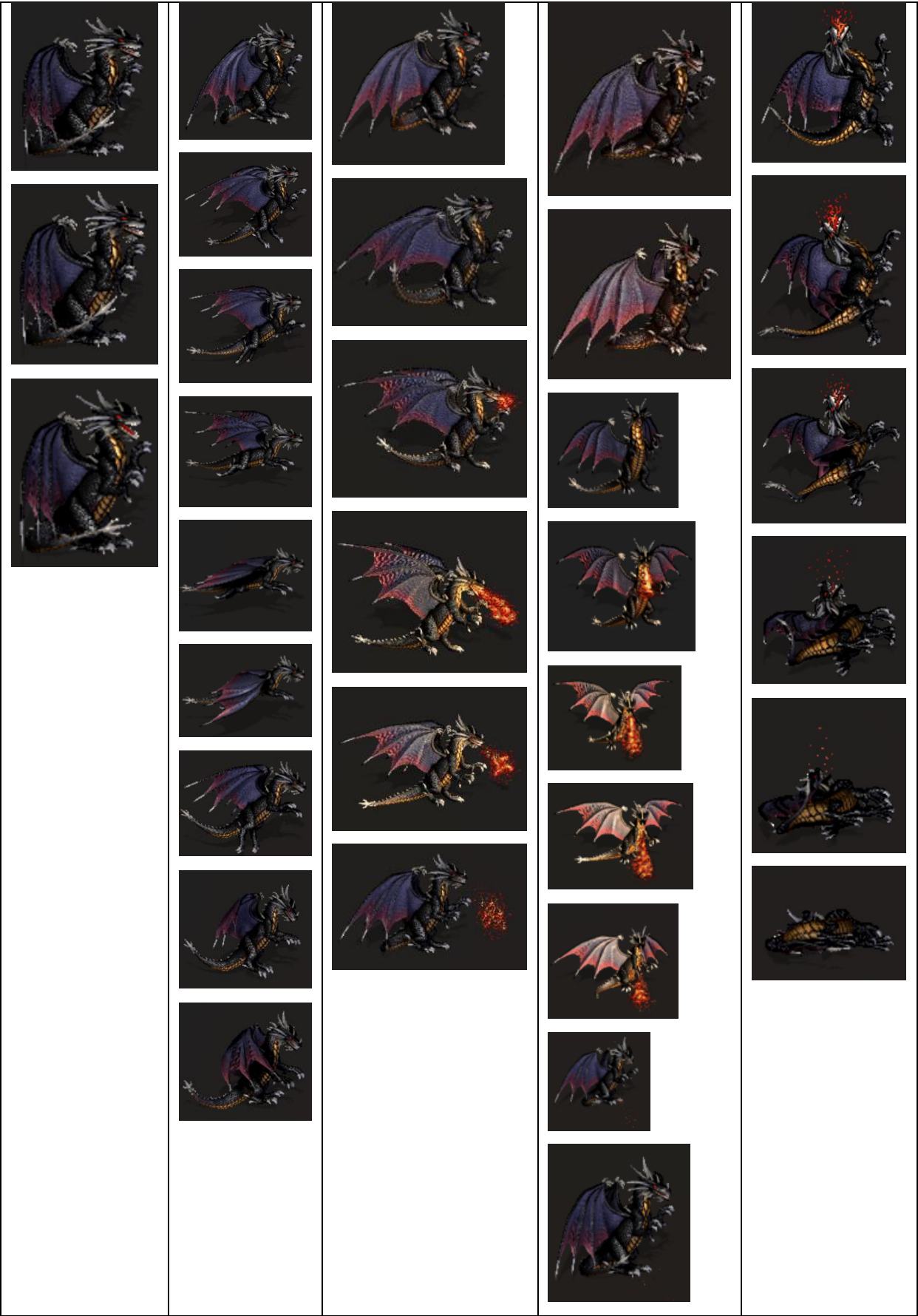
Animator





StartAnimation



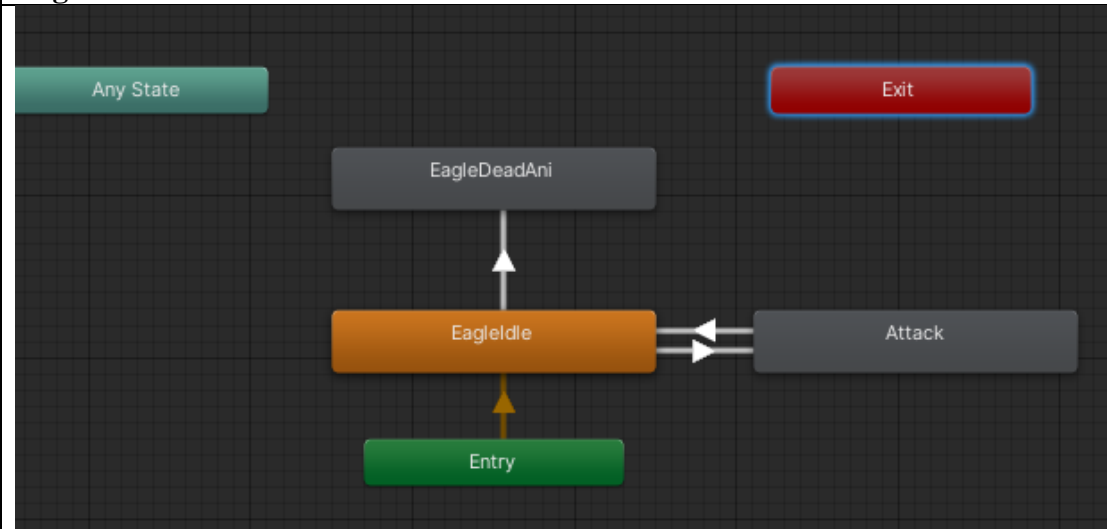




Animator

Eagle

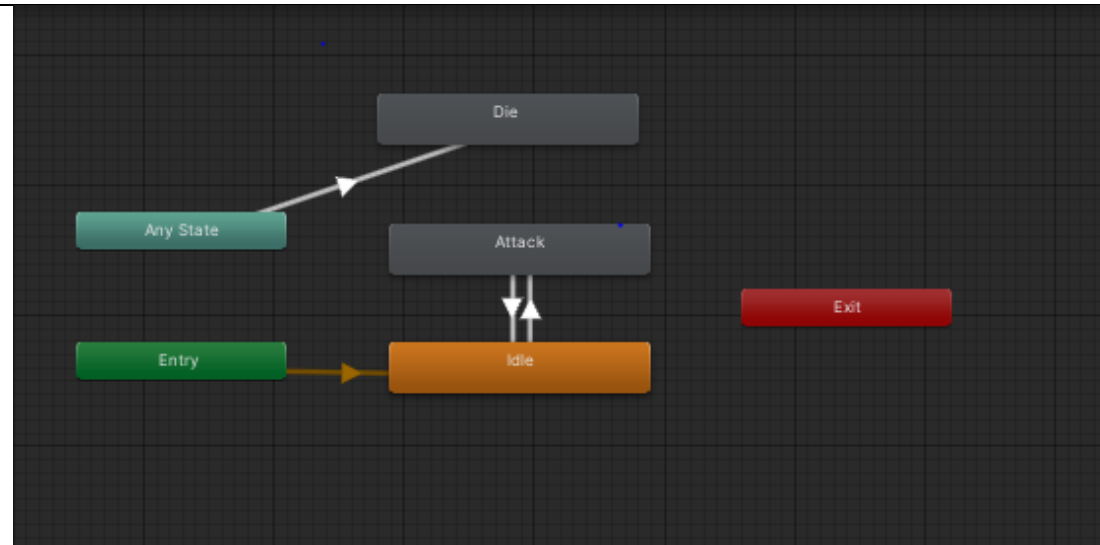


EagleIdle	EagleDead	EagleAttack
 eagle-attack-1 (40x41)		 eagle-attack-1 (40x41)  eagle-attack-1 (40x41)
 eagle-attack-2 (40x41)		
 eagle-attack-3 (40x41)		

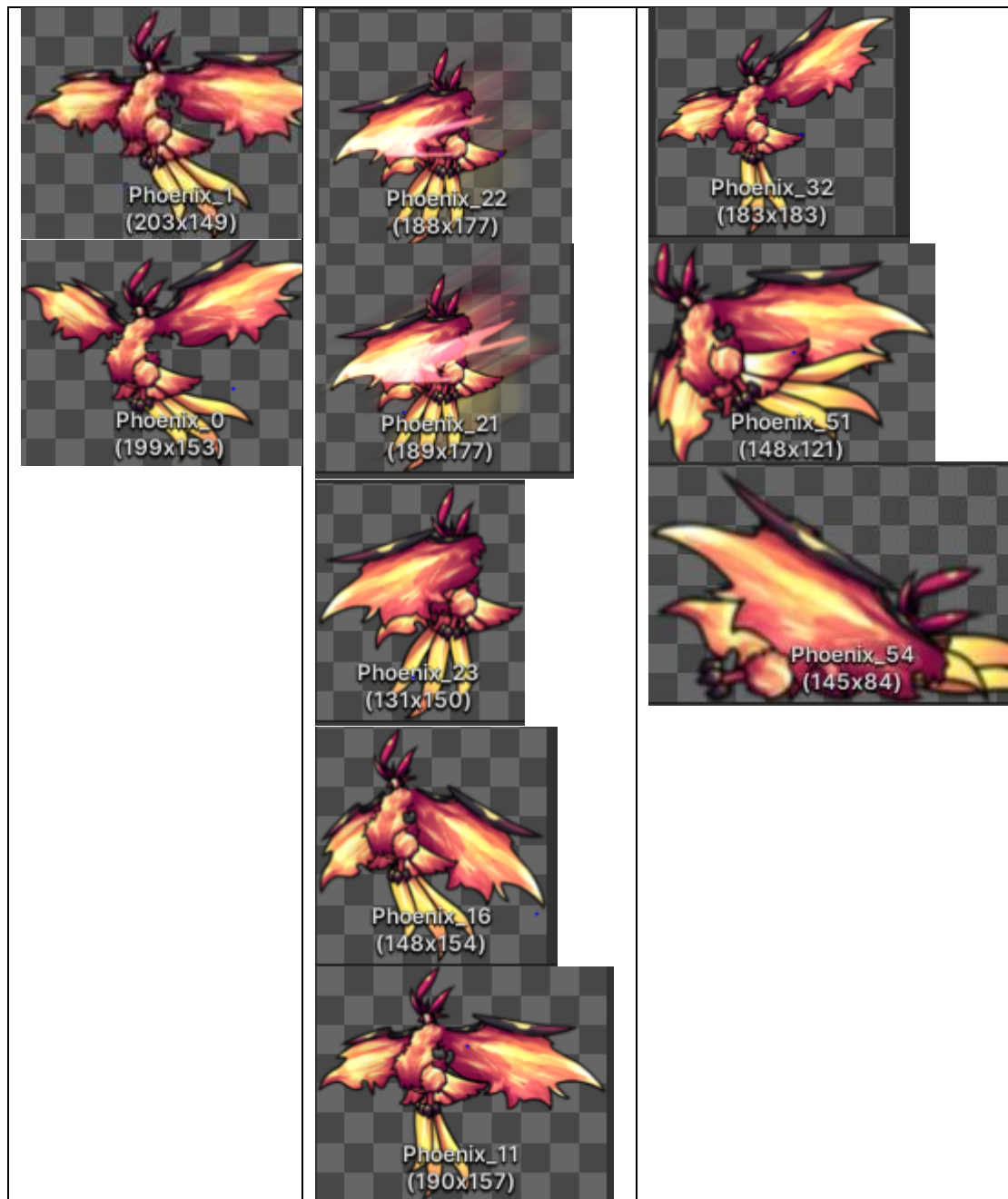
 <p>eagle-attack-4 (40x41)</p>		
---	--	--

Animator

Phoenix



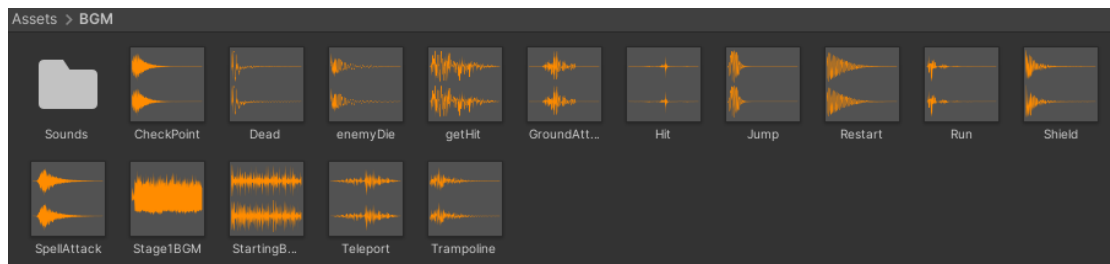
Idle	Attack	Die
 Phoenix_0 (199x153)	 Phoenix_11 (190x157)	 Phoenix_34 (187x137)
 Phoenix_1 (203x149)	 Phoenix_15 (202x146)	 Phoenix_28 (202x148)
 Phoenix_2 (149x155)	 Phoenix_16 (148x154)	 Phoenix_30 (182x183)
	 Phoenix_23 (131x150)	 Phoenix_31 (182x183)



Audio Implementation Overview

Audio File Setup (Written by Lu Jason)

In this project, all of the audio source files are store in the folder of BGM. In order to play the sound, we would apply the component of Audio Source to plug in the audio source file. Besides, we would also apply the Audio Listener so that we can hear the sound as well. The below figure is the folder of BGM in the Assets folder.



Sound Effects Overview (Written by Lu Jason)

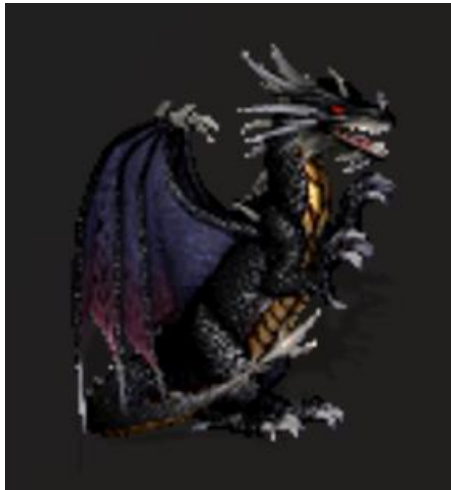
Name of the audio file	Type of the audio file	Sampling Rate of the audio file
CheckPoint	wav	2116kbps
Dead	wav	2116kbps
enemyDie	wav	2116kbps
GroundAttack	wav	2116kbps
Hit	wav	2116kbps
Jump	wav	1411kbps
Restart	wav	1411kbps
Run	wav	1411kbps
Shield	wav	256kbps
Spell Attack	wav	2166kbps
Teleport	wav	2166kbps
Trampoline	wav	2166kbps

Background Track Overview *(Written by Lu Jason)*

Name of the audio file	Type of the audio file	Sampling Rate of the audio file
StartingBGM	wav	1411kbps
Stage1BGM	wav	705kbps
Stage2BGM	wav	705kbps
Stage3BGM	wav	705kbps
FinalStage	wav	705kbps

Game AI Overview *[script]*

Dragon



- Dragon.cs, DragonBreath.cs and DragonWalk.cs are used by the dragon.
- The dragon detects the existence of player and starts to chase the player.
- Once the player enters the attack range of the dragon, the dragon starts to attack the player.
- Once the health of dragon drops below 200, the dragon will enter enraged mode and deal extra damage to the player.

- Once the health of dragon drops below 0, the dragon will die and drop a buff potion.

Phoenix



- Phoenix uses PhoenixAI.cs, SceneController.cs, EnemyPathFinding.cs, and Seeker.cs script.
- Phoenix will detect the player and chases the player.
- Phoenix will regularly drop a steel ball that hinders player character's movement on the map.
- Phoenix has a pathfinding algorithm and will avoid obstacles to move closer to the player character.
- When player character is within sight of Phoenix, it will attempt to shoot fireball bullets to the player's location.
- Once the Phoenix has reached under 50% health, it will apply a random negative buff to the player, either poisoning the player, or lowering the gravity and slowing down the player, while changing the background to indicate the type of negative buff applied.

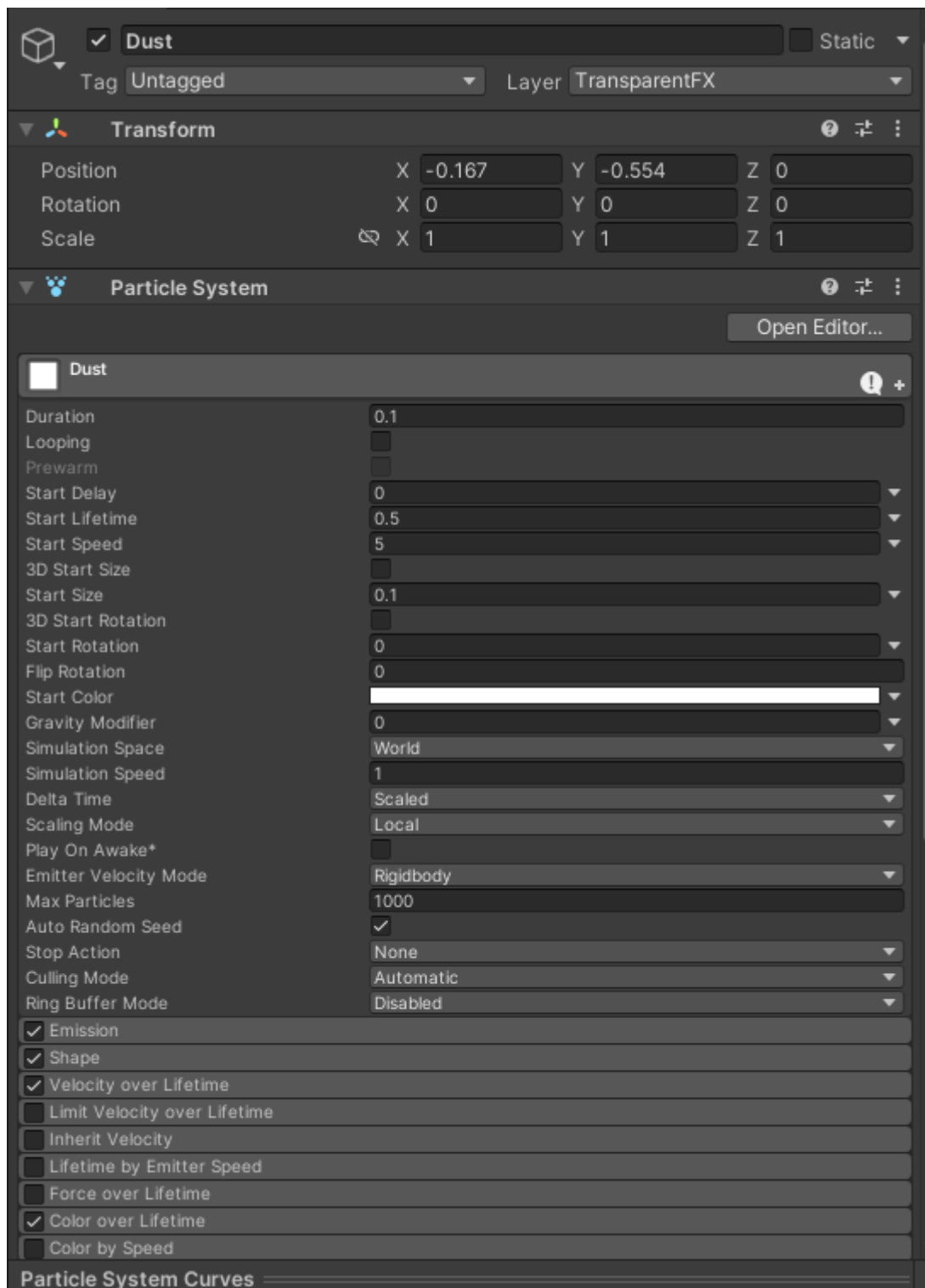
Eagle



- There are 2 types of Eagle, Eagle that follow the player and Eagle that patrol in a fix patrol path.
- Eagle use Enemy.cs, EnemyPatrol.cs, EnemyPatrolAI.cs and Chasing.cs.
- Eagle will stop and attack the player when player enter the defined area Infront of the eagle.
- Eagle will also spawn in a random location after a set amount of time.

Particles Overview






Particle System (Written by Lu Jason)

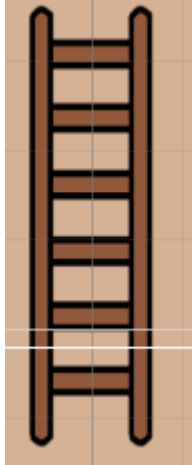
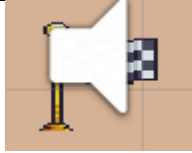
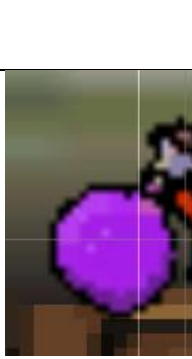
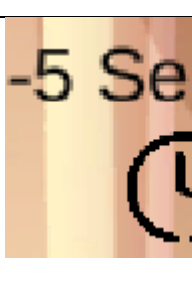



The particle applied in this project is name as Dust, it is one of the components in unity. The sprite of the dust is default, the layer of dust is same as the player layer, the start size of the dust is 0.1 so that when we play the game the dust particle would not be too

big. For the shape of the dust, we are using the box shape, so that it could present the effect of dust when user jumping. Besides, we would adjust the velocity over lifetime as well. For the colour over lifetime, we would adjust it according to the colour theme of the character. For example, the colour themes of the ninja, mage, and warrior are green, red, and yellow. Furthermore, we would also tick off the settings of looping and play on awake, since the dust only generated when the player is jumping. However, this dust effect is applied on the character of the ninja, mage, and warrior, and it is generated when the player is jumping.

Game Interactable Objects Design and Implementation (Written by All)

Game Object	Description
	<p>This is the trampoline which allows the user to jump higher.</p> <p>To let the user, jump higher, we applied the physics material 2D, and the bounciness is 1.5</p>
	<p>This iron box is allow the player to grapple on, the layer of this object is set as Grappleable.</p>
	<p>When user touch this is the portal, it would transfer the user to another portal's position.</p>
	<p>These spikes would hurt the player when the player touch it. Besides, when the player touch theses spikes, they would bounce off as well.</p> <p>To let the user bounce away, we applied the physics material 2D, and the bounciness is 1</p>
	<p>This is a moving platform to help the user pass through the traps which is underneath.</p> <p>When the user stays on this platform, they would become part of the child object of this platform.</p> <p>Hence the user would be moving together with this platform.</p>

		<p>This is the ladder which allow the user to climb on it. When user climbing this ladder, the gravity scale of the user object is 0 so that they would not fall down.</p>
		<p>This is the ending flag, when user touch this flag, they would redirect to the scene of Select Character.</p>
		<p>This is the buff potion dropped after the death of dragon. When player collides with the buff potion, the player will be redirected to the power-up scene.</p>
		<p>This is a time pickup available in stage 3. When player collides with the object, the time remaining of the stage will reduce by 5 second.</p>
		<p>This is a heal zone available in stage 4 fight against Pheonix. When player enters the area surrounding this symbol, player's health will slowly heal to the max.</p>

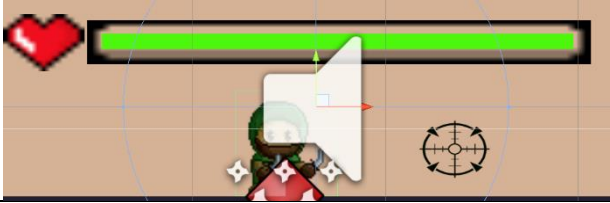
Game Interface Setup and Implementation

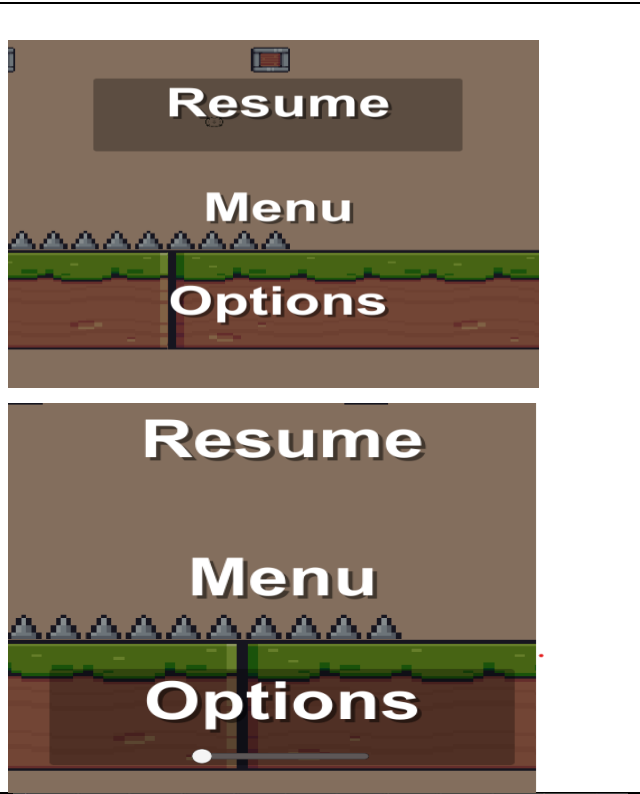

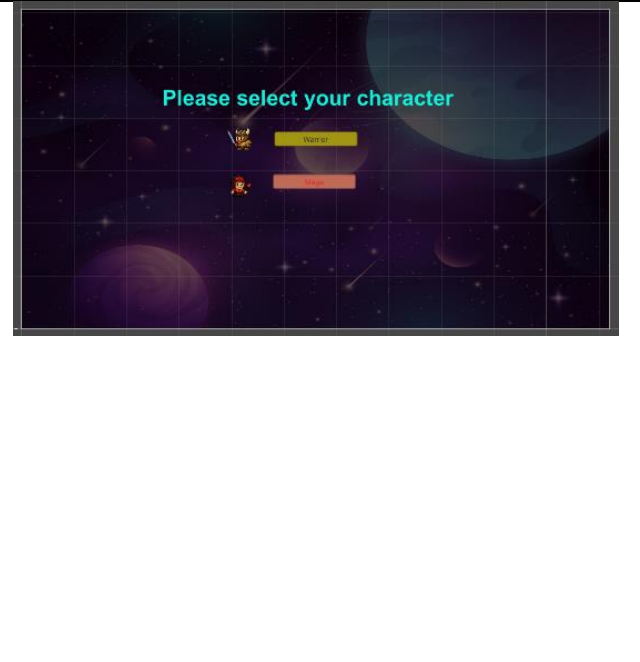
Camera[s] Setup and Implementation (Written by Lu Jason)

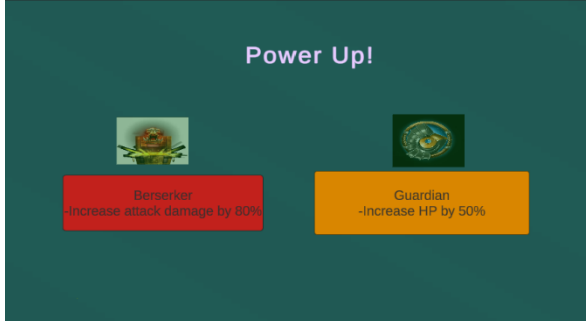

In our project, all of the camera in each scenes applied the script of CameraFollow. In this script, initially it would find the target object which is player. After the target is found, it would update the position of the camera same as the position of player in the function of Update(). Besides, we can also adjust the speed of camera to follow the player, the shake duration, the shake amount, and decrease factor as well. However, this script is attached in the object of Main Camera in each scene, and user can adjust the settings of the camera for each scene.



Canvas and other GUI Setups and Implementations (Written by Lu Jason)

Canvas Object	Description
Health bar inside the Player object 	The health bar slider represents the current health of the player. It would adjusted when the player is taking damage.
Menu	This is the pause menu canvas, when the user pressed on ESC button, this canvas would display. When user clicks on the Options in the menu, the volume slider would display, and user can adjust the volume as well.

	
	<p>This is the canvas of the start scene in this canvas it contains 1 button which is Start button. When user clicks on this button, the game would start.</p>
	<p>This is the canvas of the Select Character scene, in this canvas it contains 2 buttons, which are Warrior and Mage buttons. If user clicks on Warrior button, then they would play as warrior character for the rest of the game. Else if user clicks on Mage button, then they would play as mage character for the rest of the game.</p>

	<p>This is the canvas of warrior power-up. If the character chosen by the player in SelectCharacter scene is warrior, the player will be redirected to this scene after colliding with the buff potion in stage 2. The canvas contains two buff icons and two buttons with the descriptions. The player needs to choose a buff and clicks on the button of the buff desired to power-up the warrior.</p>
	<p>This is the canvas of mage power-up. If the character chosen by the player in SelectCharacter scene is mage, the player will be redirected to this scene after colliding with the buff potion in scene 2. The canvas contains two buff icons and two buttons with the descriptions. The player needs to choose a buff and clicks on the button of the buff desired to power-up the mage.</p>

Game Options Setup and Implementation (Written by Lu Jason)

The game is started with the scene of Start scene. In this scene, user can click on the Start button to start the game. When the game is started, user can press on the ESC button to open the pause menu canvas.

In the pause menu, it contains 3 buttons, which are the buttons of resume, menu, and options. To resume the game, user can click on the resume button, to return to the start scene, user can click on the menu button. Besides, user can adjust the overall volume of the whole game by clicking on the option button. When the option button is

clicked, the volume slider will display, this slider represents the current overall volume of the whole game.

Project Development Progression

Project Schedule (Written by Lu Jason)

Milestone 1

1. Decide the type of the game, story of the game, characters of the game as well.
2. User movement
 - using key such as A and D to move left and right, and space to jump, double space for double jump, SHIFT for dash.
3. Search for the assets of the characters, backgrounds, enemies, and items.
4. Design the ground/platform.
5. Develop the first version of scene of start scene, stage 1 scene, select character scene.
6. Setup the source tree as an integration tool for our project.
7. Game Logics
 - Whenever user get hit by the enemy/trap, the hp will decrease.
 - User can walk, run, dash, jump and double jump.
 - When enemy get hit by the user, the enemy hp will decrease as well.

Milestone 2

1. Develop the scene of stage 2, 3 and final.
2. Modify the scene of start scene, stage 1 scene, and select character scene.
3. Run all the scenes, fix any detected bugs in each of the scene.
4. Develop the AIs of stage 2, 3 and final.

Milestone 3

1. Combine all the scenes.
2. Polish the code, to make it more quality.
3. Rerun all the combined scenes, fix any detected bugs in each of the scene.
4. Start to write the final report.

Progress Report (Written by Lu Jason)

Week	Activities conducted
1	Grouping
2	Discussed about the game type, story, and characters of the game.
3	Complete and submit the proposal report
4	Develop the object of characters, and the scene of stage 1
5	Develop the scene of stage 2, 3 and final.
6	Presentation of milestone 2
7	Develop the ai of stage2
8	Develop the ai of stage 3
9	Develop the ai of final stage
10	Presentation of milestone 2
11	Polishing the code.
12	Fix the bugs detected in the code.
13	Milestone 3/ Final Milestone presentation
14	Complete and submit the report and project

References [*in APA format*]

Rawat, A. (14 April, 2021). *Case Styles in Programming*. Retrieved from system weakness: <https://systemweakness.com/case-styles-in-programming-b4ee6012fd5f>

Unity documentation. (n.d.). Retrieved from Unity: <https://docs.unity3d.com/2020.1/Documentation/Manual/system-requirements.html>

Unity Learn. (7 December, 2021). Retrieved from learn unity: <https://learn.unity.com/course/create-with-code>