Assignment One

Unity Game Development

By: Alisa Ordina

ID: 100967526

Table of Contents

1. [Detailed game description 2](#_Toc497522694)
2. [Description of game’s controls 2](#_Toc497522695)
3. [Interface Sketch 2](#_Toc497522696)
4. [Screen descriptions 3](#_Toc497522697)

[Scene One](#_Toc497522698)

[Scene Two](#_Toc497522699)

[Scene Three](#_Toc497522700)

1. [Enemies 6](#_Toc497522701)
2. [Game objects in the game 7](#_Toc497522702)

[Star](#_Toc497522703)

[Red Bird](#_Toc497522704)

[Cloud](#_Toc497522705)

[Red Explosion](#_Toc497522706)

[Blue Explosion](#_Toc497522707)

1. [Scoring 10](#_Toc497522708)
2. [Sound Index 10](#_Toc497522709)
3. [Art and Multimedia index 11](#_Toc497522710)
4. [Sources 12](#_Toc497522711)

[First Source](#_Toc497522712)

[Second Source](#_Toc497522713)

1. [References 13](#_Toc497522714)

# Detailed game description

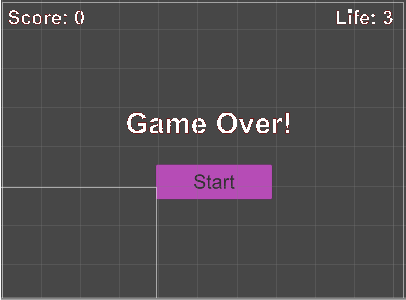
This game is about the main player who is a Red Bird. He is a hero and a legend in the Society Of Birds. However, he likes to fly in forbidden places, including the Village Of Man. Since humans hate this bird so much they throw bowling balls, the only things they own. Since he is not an ordinary bird he loves to fly in dangerous places. The user should avoid the bowling balls by flying between them to catch the stars and fly safely over back to his family. Collect the most stars to earn bragging rights. The higher your experience points the higher your score.

# Description of game’s controls

The user could control the Red Bird avatar by using a keyboard with specified keys as follows: pressing on W key: is to go up, pressing A key: to go to the left, pressing S key: to go down, and pressing D key: to go to the right. Now how this avatar could avoid these harmful throwing objects? It is simple, there are two ways: one: it is done by simply avoiding these objects by using AWSD keys, and by flying around them. The second option it is done by hitting a space bar key on the keyboard. The space bar allows the red bird to fire and could destroy these flying objects when aimed properly. Where ever the Bird is facing that is where the fire will go. So, aim and fire right at the specific object, but careful not to collide with it. This could deduct the Bird’s life points. However, if the bird’s life has been depleted, the User Interface/Head Up Display will be displayed. The user can only click on Star button to restart this game again.

# Interface Sketch

The GUI (graphic user interface) was created from the UNITY’s Game Object UI (user interface) tools. So, by using Unity’s HUD (Heads Up Display) I created a user interface with specific colors and fond sizes. This UI is represented on the following figure below.



After creating Head Up Display and customizing fonts and its colors in Unity. I renamed all the labels as follows: score label: scoreLabel, Life lable: lifeLable, Game Over label, gameOverLabel and Start button: resetBtn. As shown in figure below all the renamed labels that are contained within the UI canvas hierarchy.



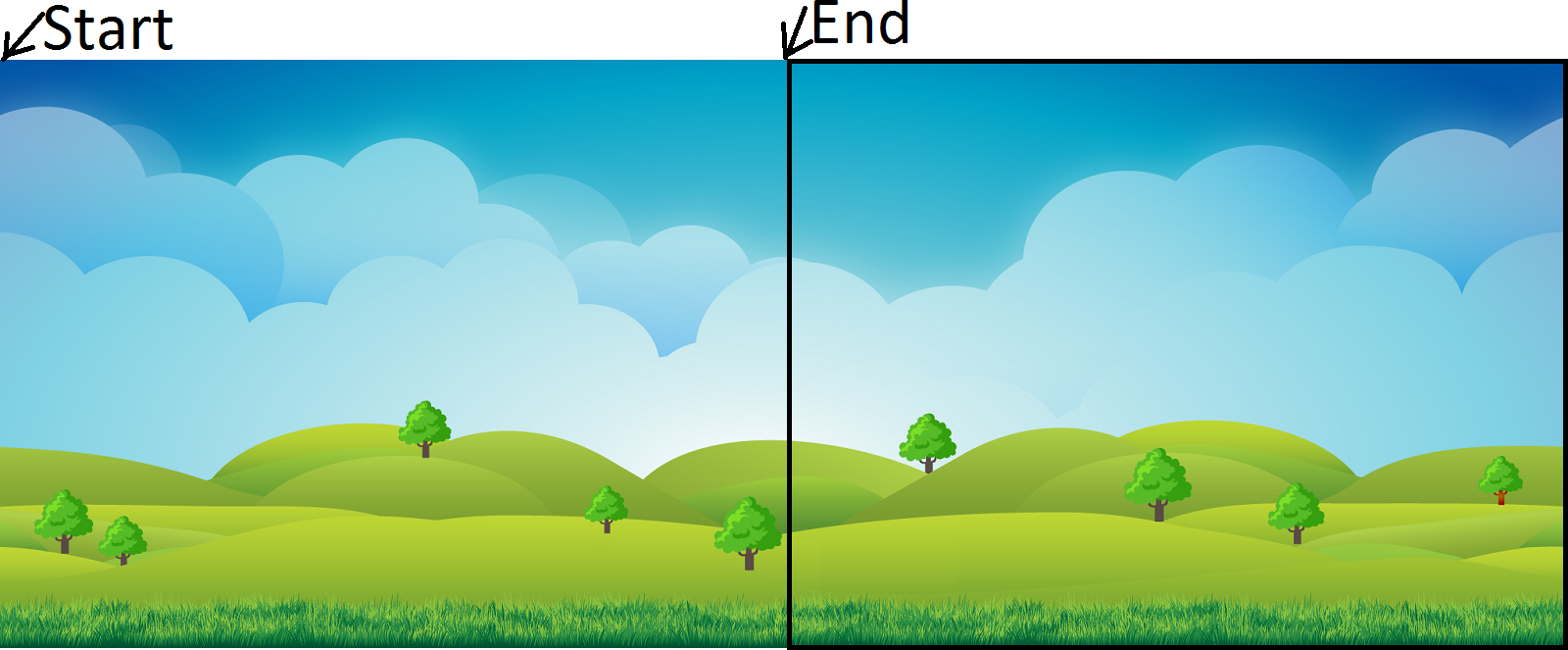
Then script called HUDController is created which will update the score and life label when appropriate in the game and display Game Over! label with a Start button when the bird’s life has been depleted. This script is attached to the Canvas and the canvas’ labels and button event. The HUDController also has association with the Player class. The Player class which is a singleton, is a special class that only could be a single copy with in this system. Meaning that if this game had multiple levels, the Player class could be carried over to another level. The Player class interacts with the HUDController, because the Player class keeps track of score and life points of the main Bird player.

# Screen descriptions

I was looking for just one good background for this game, preferably environment background. I came across this website: <https://opengameart.org/content/3-parallax-backgrounds>. This is the website where I found some interesting backgrounds. I have only downloaded one of them as shown in the figure below.

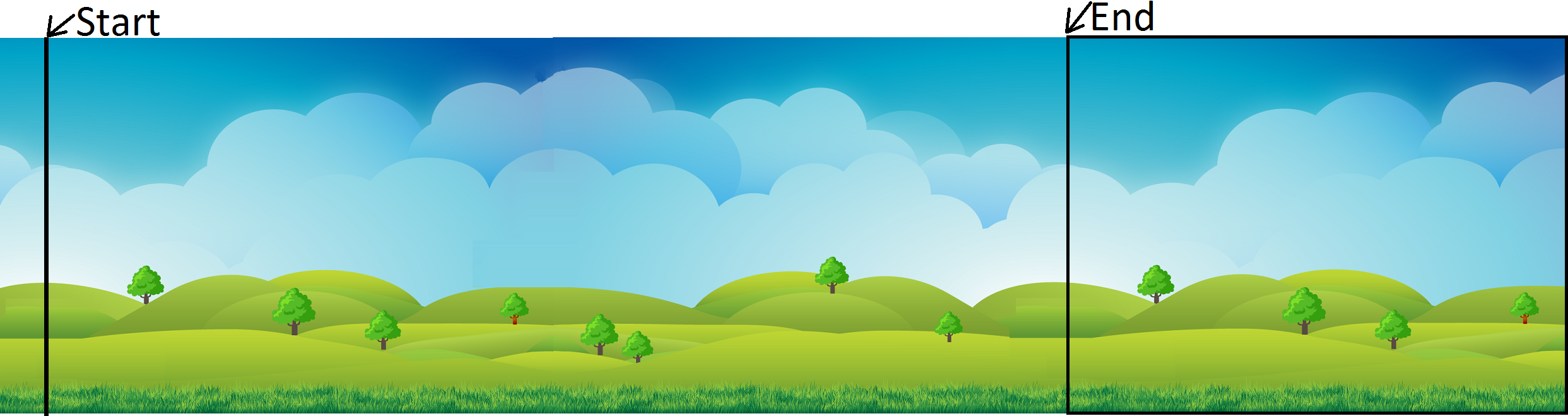


After importing this new asset into Unity, for my game’s background. I must create a script that will apply appropriate transformation to this picture to move it smoothly horizontally in (x axis). This script is called BackgroundScroller which consists with the background’s speed and the x axis boundary. The x axis boundary should have a x start point and x end. After this script was created and attached to the background asset, this background was jumping a lot when it was reset back to the start x axis point. Below shows the start and end of background within the camera view.



So, I had to adjust the background picture to make the background scrolling smoother. Below shows the adjusted background picture.

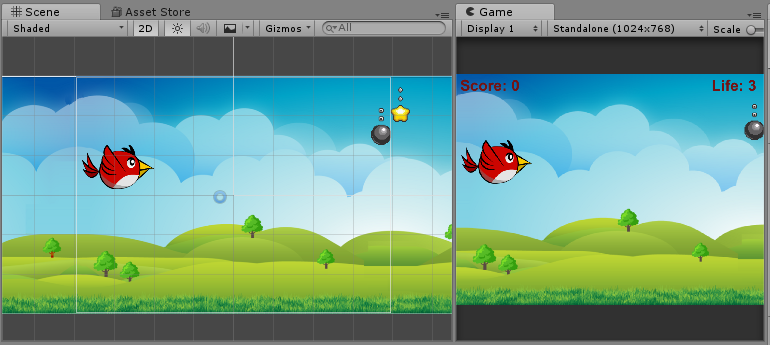




After I fixed the background asset with its script attached. Now this background has a proper x axis boundary where it resets smoothly with its speed.

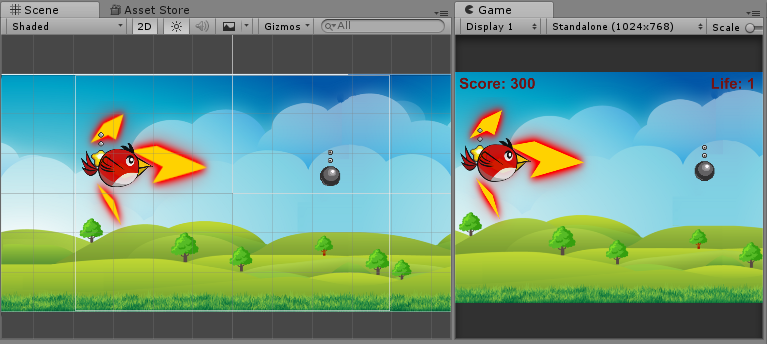
## Scene One

I placed the main Red Bird player onto the scene. Since I wanted the main player to animate always the on wake is set to true and loop in animation window is set to true. Once executed the enemy object become active and shows up randomly with its random speed on the camera view. The star (with bonus points) moves with the same speed constantly but appears at the start of x axis with random y point. This way the avatar is challenged by enemy, but it could still get to the star by knowing where roughly it will be hovering. Once the execute Play button in Unity is pressed the first scene is shown below.



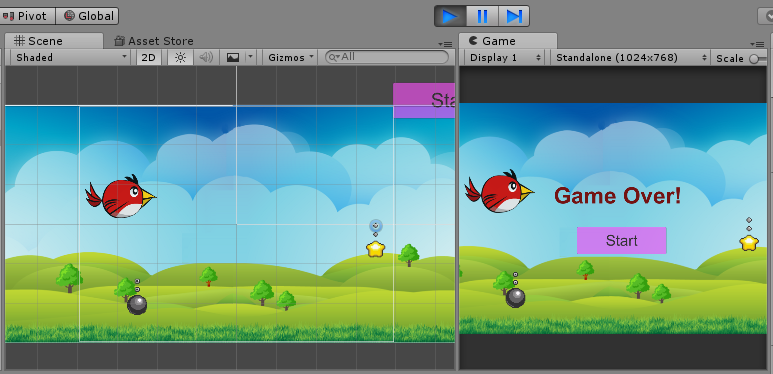
## Scene Two

The second screen shot is when the Red Bird has been hit with enemy (bowling ball), but gain 100 points for hovering over the star. Because the Red Bird has been hit by the enemy, the player’s life has lost 1 health point.



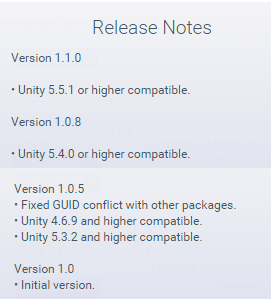
## Scene Three

This is the last scene when the Red Bird’s life has been depleted. The user could press the Start button if he/she would wish to play this game again. As mentioned before when the bird’s life is depleted, the UI or Heads Up Display will be displayed to the user. Here is where the user will make an ultimate choice whether to play this game again.



# Enemies

The enemy as described in the game’s story above, is a bowling ball. This bowling ball should be thrown at the Red Bird. I first found this game object asset in Unity store under name: Sprite Pack #1 – Tap and Fly version 1.1.0. The release notes are described in the figure below.



From this pack I downloaded the enemy sprite, the star sprite, cloud, and the sound audio for this game. The enemy is similar image as the bowling ball as shown in the following figure.



After this asset image was deployed under my assets folder, I added it to the scene and added few components that this game object needed. First this game object needs a Physics 2D Circle Collider to detect the collision with another game object. The circle collider is the best shape matched with this object. Since the Red Bird game object is not using any physics nor forces just using transform with hard coded keys inputs, the enemy’s circle collider’s trigger is set to true. This is to detect when the game object as entered with other object for the function OnTriggerEnter2D to be invoked. So, the collider is set to trigger for Unity to calculate and invoke the specific function for it to be active when two objects are detected in the specific axis point where they intercept. To make enemy move at random speeds, the minimum and maximum speed boundaries are set in the EnemyController script. The minXSpeed is set to -2 to move enemy in opposite direction. The x and y axis boundaries are also set for this enemy object to reset itself properly. The y axis is set randomly within y axis boundary to enemy appear randomly in the y axis in the scene. This way the user could be challenged with this enemy object. The Audio source component was added to this game object and was also added in the script. The script will be executed OnTRiggerEnter2D where the sound will play when appropriate. After that the enemy’s Tag has been created called “enemy” in order to compare the tags and apply it to the on trigger enter Collison function. This enemy object is also instantiated (created) again within 30 to 60 seconds onto the scene this is created in coroutine “AddEnemy” in the HUDController script. This is meant to challenge the user. After this script is attached to this enemy game object, audio source component is added with the sound, and after this game object is configured I saved it under prefabs folder for Unity to recycle this game object when needed.

# Game objects in the game

## Star

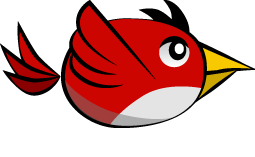
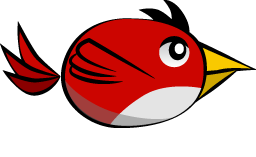
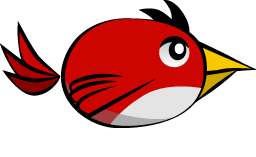
The star game object as described in the game’s story above, gains the Red Bird’s experience points. This game object was obtained from Unity’s store from Sprite Pack #1 – Tap and Fly version 1.1.0. The star game object example is shown in the following figure below.



The star’s script is called StartController which looks like background scroller script. However, the star script differs because the star appears randomly in y axis but has constant speed for a user to have a chance to gain the star game object’s points. Because we need to identify if this was in fact a star game object the star’s Tag has been added to this game object. Another component to the star object was added called Physics2D Circle Collider. The circle matches the best shape for this star object and the trigger was set to true. After these items were added, this script is attached to the game object and after the final configuration was done, this game object was saved under the Prefab folder.

## Red Bird

The Red Bird game object as described in the game’s story above, is the user’s avatar. The Red Bird game object could be controlled by the user by using keyboard with WASD indicated keys. This game object was obtained from Unity’s store called 2D Game Started Assets, version: 1.1 which is compatible with Unity 5. I selected all frames for the Red Bird game object to animated while the game is executed. The Red Bird game object example is shown in the following figure below.

1.2. 3. 4. 

When adding this Red Bird game object onto the scene, I know that this would be an avatar pf the user. That is why more controls should be added onto this game object. This script is again having similar functions such as having access to game object’s component transform, position, and its variables. However, this script that is called BirdController also contains Input which is a class defined in Unity API as a player’s input. This could vary, but for this game I am using keyboard keys. After Input is initiated the keys are indicated as well, which would be pressed by the user. When this occurs, and the user’s key is registered the specific state would be invoked. After the key event passed and will return true, only while the user holds down the key. The Red Bird game object will behave according to the specific key that was pressed. This would be either W: up, A: left, S: down and D: right. Then the movement of this game object is applied by using Vector2 for the two-dimensional axis to move the Red Bird vertically or horizontally in the scene. The boundaries are applied in y and x axis for this game object to not appear off from the camera view. Another input was created which is space input on keyboard that would instantiate create a cloud game object. This is when the Red Bird would fire at the opponents. This is done by getting the Red Bird position and instantiating the cloud without any rotations. Because this object does not use the forces and physics the kinematic rigid body component is attached to this game object. The Polygon Collider2D is also added to detect the collision. Because of this bird’s weird shape, the polygon collider is the best fit. The audio source component is also added on to the Red Bird game object. The Bird Collision script is also created to detect the collision between objects in OnTriggerEnter2D function. Here is where the red explosion would be instantiated (created) onto the scene, and the life subtracted when the collision between Red Bird and enemy would happen. However, if the collision would have happened with the star the following sound would be triggered, and point would be added on to the user’s score. In the Bird collision script contains coroutine would also be called to change this game objects color when collision with the enemy has happened. To change the game object’s color the material accesses this game object’s color over this game object to change the game object’s invisibility. This is done with alpha channel when collision with the enemy happened the alpha channel will be set to 1 and back to normal. This means when the collision occurred with the enemy this game object will blink. After attaching all the red Bird’s scripts and configuring this game object I saved under the Prefab folder.

## Cloud

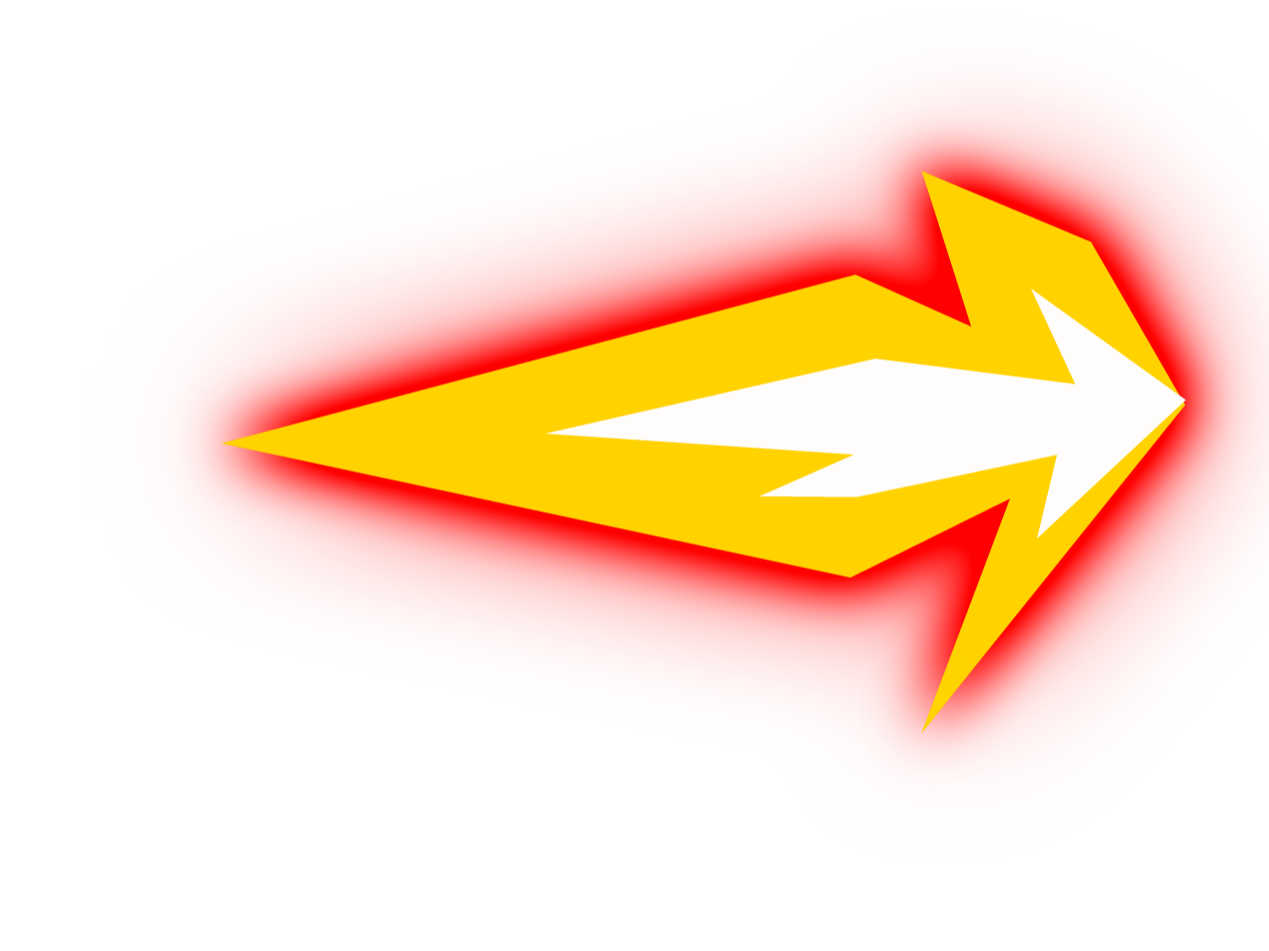
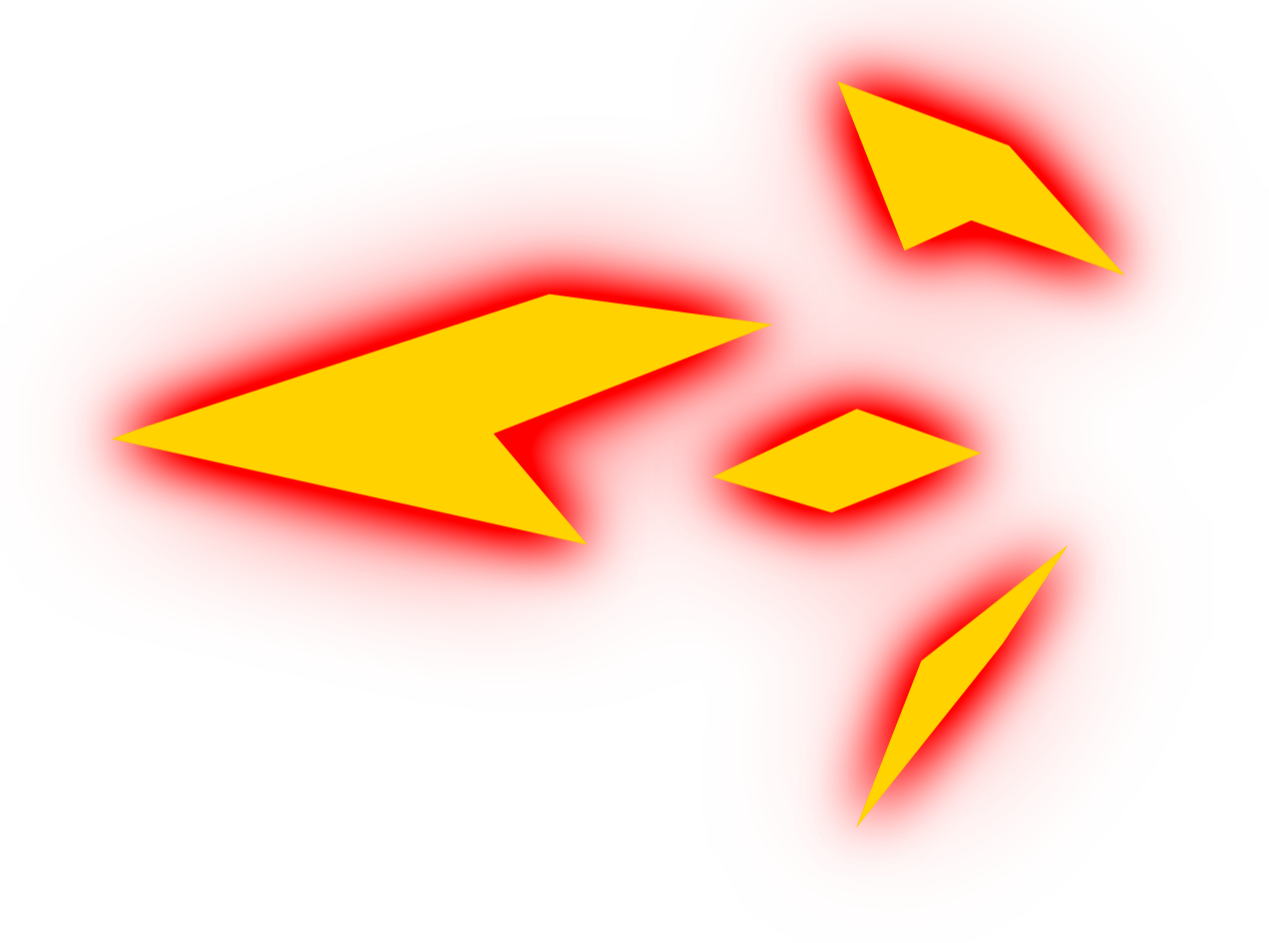
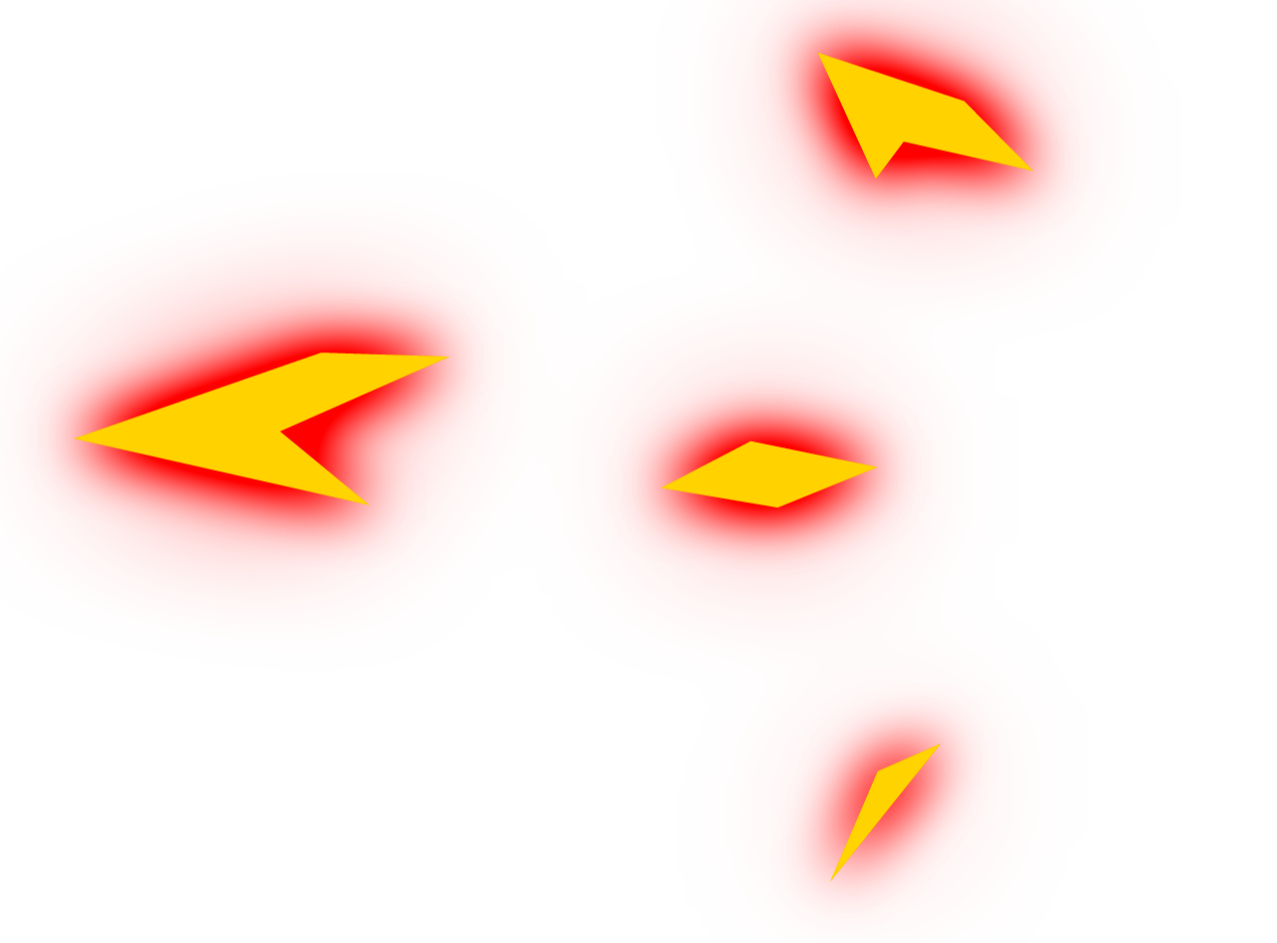
The cloud game object is the Red Bird’s fire. This game object was obtained from Unity’s store from Sprite Pack #1 – Tap and Fly version 1.1.0. The cloud game object example is shown below.



The cloud script is called CloudController which is like starController. However, the cloud script differs because the cloud game object does not show up in random y axis, instead it instantiated where the Red Bird’s position is defined. However, the cloud object is reset when it either hits the enemy or the x axis boundary of the scene. Because we need to identify if this was in fact a cloud game object the cloud’s Tag has been added to this game object. Another component to the cloud object was added called Physics2D Polygon Collider. The polygon best matches the shape for this cloud object and the trigger was set to true. Also, a rigid body was added to this game object to collide with the enemy object which is set to Kinematic. After these items were added, this script is attached to the game object, the audio source is attached, and after the final configuration was done, this game object was saved under the Prefab folder.

## Red Explosion

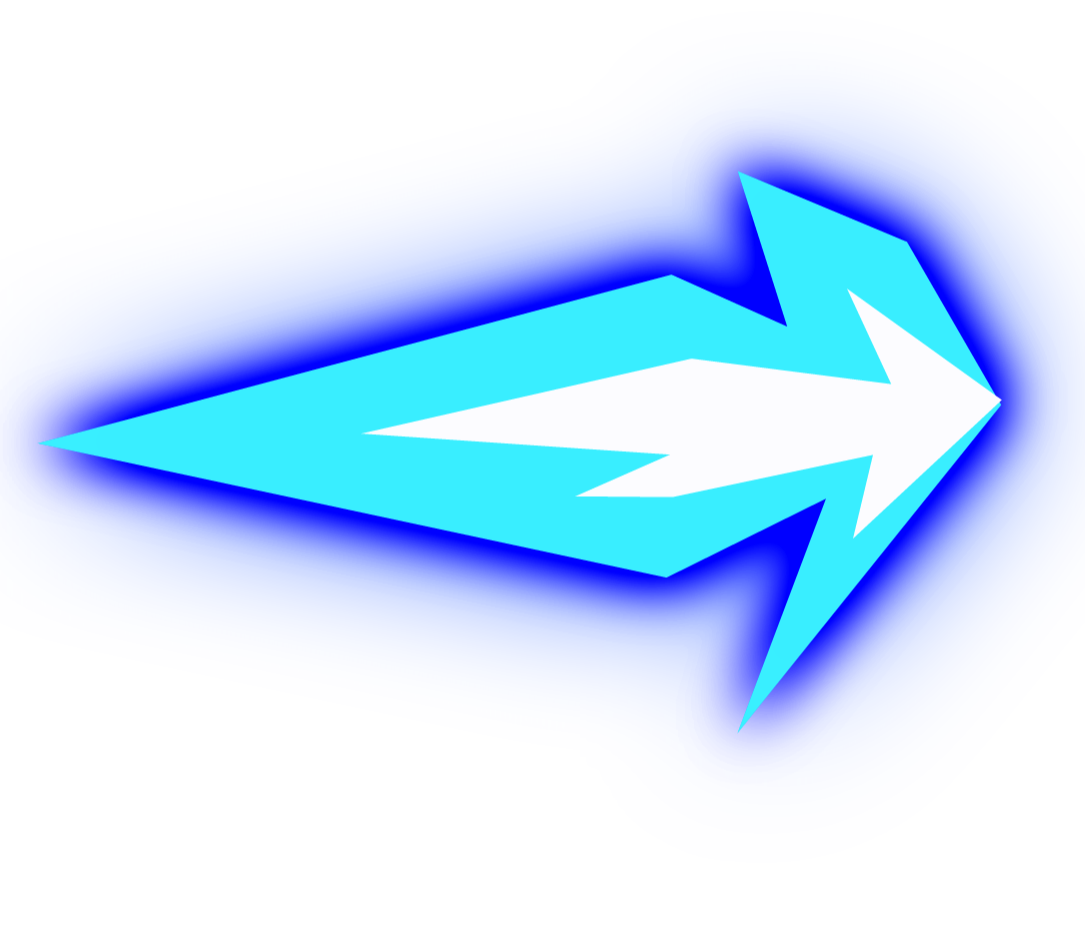
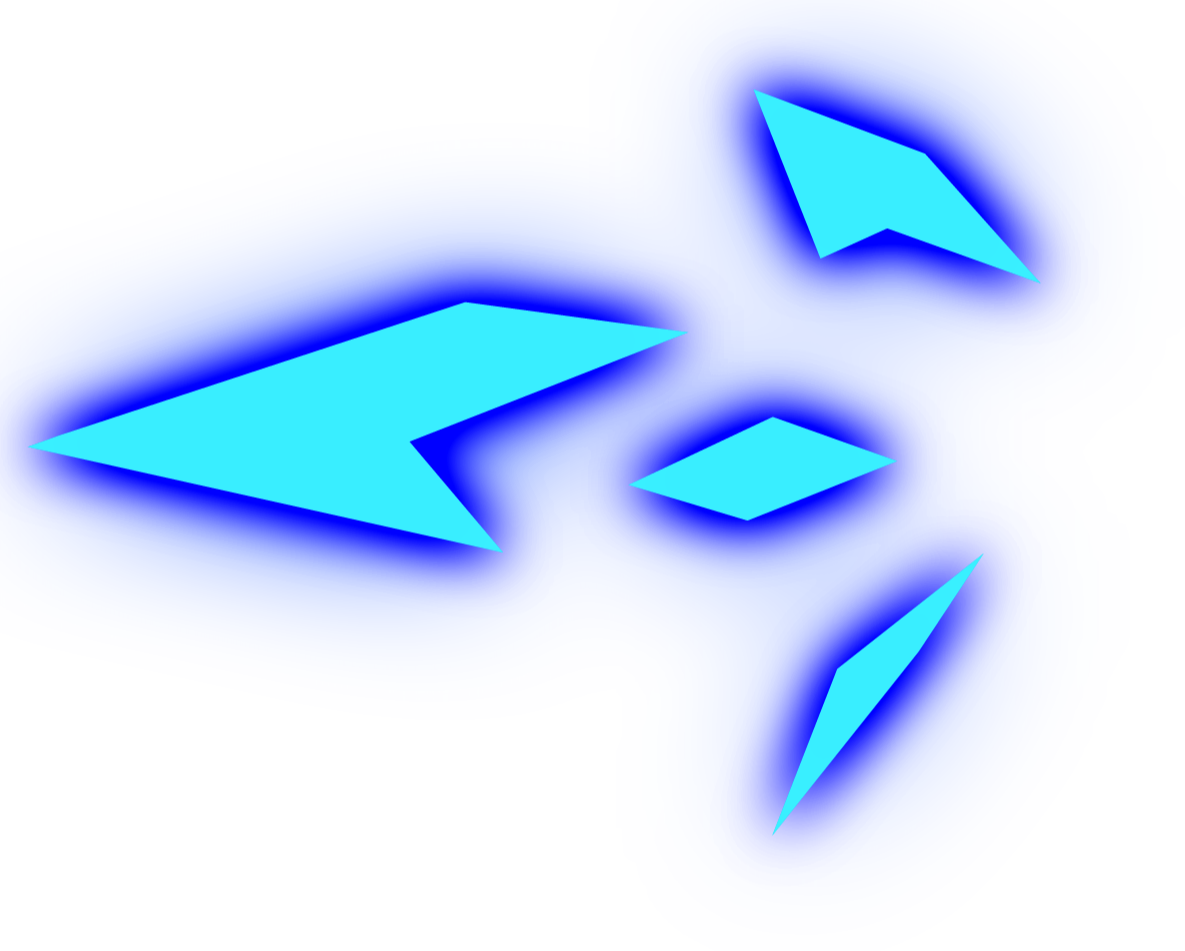
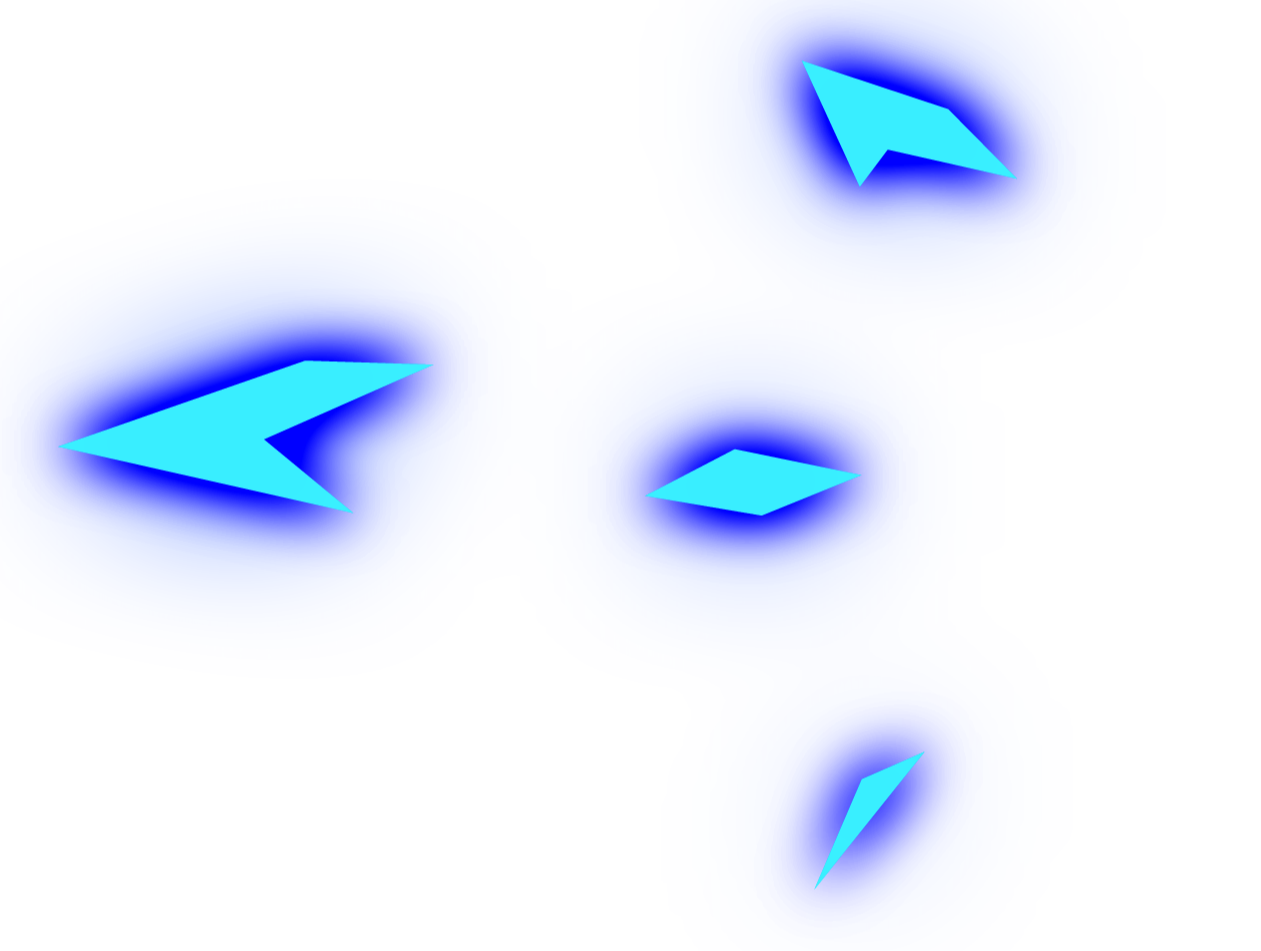
The Red Explosion game object is the Red Bird’s collision with the enemy. This game object was obtained from Unity’s store from Sprite Muzzle Flashes 1.0. This is the first release of these assets. The example of this explosion displayed below.

1.  2. 3. 

This game object was selected with all frames and saves under the animation folder. Because I don’t need this animation playing repeatedly the Loop Time is set to false not checked. While this game object in on the scene I also added an extra component, which is script that has a function that destroys this game object. After that I accessed the animation window and added the destroy function after the last frame section. This means that this game object will be invoked at the collision and will disappear (destroyed) in the scene. After this game object was fully configured I saved it under Prefab folder and deleted it from the scene.

## Blue Explosion

The Blue Explosion game object is the Red Bird’s fire collision with the enemy. This game object was obtained from Unity’s store from Sprite Muzzle Flashes 1.0. This is the first release of these assets. The example of this explosion displayed below.

1. 2.3.

This game object was selected with all frames and saves under the animation folder. Because I don’t need this animation playing repeatedly the Loop Time is set to false not checked. While this game object in on the scene I also added an extra component, which is script that has a function that destroys this game object. After that I accessed the animation window and added the destroy function after the last frame section. This means that this game object will be invoked at the collision and will disappear (destroyed) in the scene. After this game object was fully configured I saved it under Prefab folder and deleted it from the scene.

# Scoring

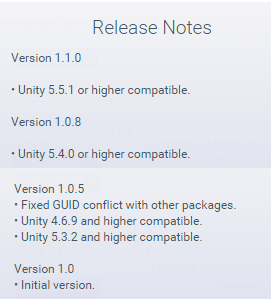
First the HUD (Heads Up Display) is created or Unity’s UI. After creating all the components or the children of UI Canvas the score label, life label, Gave Over! Label and button. The script is created called HUDController. In this script the UI and SceneManagement classes are imported (using) to use (access) the UI labels and buttons and switch to active scenes when the button in UI is pressed. After update UI and Game Over function are creating to update the User Interface screen in the scene, the Player class is also created. The Player class is a singleton class that could have one copy with its entire system. This is perfect because if this game consisted with multiple levels the Player class could easily be passed on to the next level without creating another duplicated copy of the Player class. The Player class that is a class on its own without inheriting the MonoBehaviour tracks the score points and life point with communicating it to the HUDController class. However, because the Player’s class property is public the HUDController could access them. So, when the Red Bird player passes a star, 100 points will be added on to the current score point. Collect the most stars to earn bragging rights. The higher your experience points the higher your score.

# Sound Index

The audio source components that are used are named, Score, Hit, Flap that are from in Unity store under name: Sprite Pack #1 – Tap and Fly version 1.1.0.

| Sound Name | Audio Source |
| --- | --- |
| Score | This audio source component is added into the Red Bird gameObject. This sound is played when the Red Bird flies over the star. This collision triggers this Ding sound to let the user know that they have gained points. |
| Hit | The audio source component is added into the cloud gameObject. This sound is played when the Red Bird fire at the enemy. If the cloud hits the enemy the hit sound would be played again. To let user know that they have hit the enemy. |
| Flap | The audio source component is added into the enemy gameObject. This sound is executed when the Reb Bird gets hit by the enemy. This hit sound is invoked with the enemy collision to notify the user that their avatar has been hit and it lost a life point. |

The release notes are described in the figure below.



The background main camera’ audio source component called music\_heroic\_background is from Unity store under name: Variety Pack 01 version 1.0 which is their first release.

| Sound Name | Audio Source |
| --- | --- |
| music\_heroic\_background | The audio source component is attached into the camera gameObject. This is the background music for this game that loops over again. |

# Art and Multimedia index

This is the list of all game objects that was used in this game Development. Please see the following table below.

| Asset’s Name | Assets |
| --- | --- |
| Red Bird | 2.3.4. |
| Star |  |
| Enemy |  |
| Cloud |  |
| Red Explosion | 2. 3. |
| Blue Explosion | 2.  3. |

# Sources

## First Source

The lab COMP 3064 and its youtube video source recording of this lab, helped me to make this game. The few ideas came mostly from our prerecorded labs, because it helped to understand the theory behind the code and most important how to configure the game object.

https://www.youtube.com/watch?v=1Yv\_LazuNR0

## Second Source

The second source which help me make more interesting game object. The Red Bird’s fire against the enemy had to be instantiated. This ideally should make the enemy object to disappear to make Red Bird’s pathway more freely to fly along. This source I got from our Assignment One description it called: Unity3D - Space Shooter Tutorial.

Its under name of: c# Unity3D - Space Shooter Tutorial Series: 01

From: <https://www.youtube.com/watch?v=48anhmWxT54>

# References

Przemyslaw Pawluk. 2017, September 15. COMP3064-F2017-Week2. Retrieved from

<https://www.youtube.com/watch?v=1Yv_LazuNR0>

Jonathan Weinberger. (2013, March 27). c# Unity3D - Space Shooter Tutorial Series: 01.

Retrieved from https://www.youtube.com/watch?v=48anhmWxT54