**CIS2150**

**Games Engines**

Level 5

Coursework 2: Game Project (Group Report)

2020/2021

**Individual Report**

Written by

McLaughlin Callum (24678805).

Contents

[**Introduction** 3](#_Toc60955986)

[**Contributions to the Project** 3](#_Toc60955987)

[Research 3](#_Toc60955988)

[Gun Script 3](#_Toc60955989)

[Bullet Script 4](#_Toc60955990)

[Group Report Contributions 4](#_Toc60955991)

[Testing Gun Script 5](#_Toc60955992)

[Testing Bullet Script 6](#_Toc60955993)

[**Code Listings** 7](#_Toc60955994)

[Gun Script 7](#_Toc60955995)

[Bullet Script 8](#_Toc60955996)

[**References** 9](#_Toc60955997)

# **Introduction**

The main concept around our game was that a mechanic would be implemented to allow the player to control the direction of gravity in the game. Allowing them to move from the floor to the ceiling and thus creating a simple puzzle game concept where the player would have to flip the gravity the correct way in order to proceed to the next stage. E.g. if the player is standing on the” floor” and the exit to the next stage is on the “ceiling” then the player would have to shoot the gun at the “ceiling” to force the gravity onto it and thus flipping the gravity, moving the player towards the “ceiling” which would then be the new “floor”.

The gravity flipping mechanic would be implemented using a “Gun” and “Bullet” script so that when the player fires the gun using the designated control, a bullet is instantiated, and force is applied to that object to move it forward. Then when it collides with the wall, the function which flips the gravity is called and the gravity is flipped, with the bullet model disappearing shortly after impact.

# **Contributions to the Project**

## Research

In order to start creating the mechanics for the gun and bullet I started by doing a few days of research and testing different ways I could implement my ideas. The main two concepts that I came across were using either the collision method or raycasting. The collision method would involve detecting when the bullet hit another object, the gravity flip method would then be called. The raycasting way seemed a bit more complex and involved a lot more components and steps so I ultimately decided to keep it simple and stick with the physical collision detection method. I planned to instantiate a bullet after the user fired the gun and have the bullet travel towards the wall then on impact with the wall, the onCollision method would be invoked and the code inside of it would run, which would be the code to flip the gravity which Andrew would be tasked with. Ultimately I was able to complete the scripts and everything ended up going according to plan apart from a few minor errors along the way which I was able to fix with the help of my team.

## Gun Script

The gun scripts main purpose is to handle the instantiation of the bullet object at the guns position and rotation and apply force to it, allowing it to move forward like a bullet. In the update method, there is an if statement to check if “mouse0” is being click, if it is then the force is applied to the instantiated bullet if “Time.Time > fireRateTimeStamp”, which is just a timer to handle the fireRate of the gun.

The FireGun method is what instantiates the bullet object when “mouse0” is clicked, a gameObject is created and the bullet prefab is set to it, the bullet is then instantiated at the position and rotation of the gun using the transform methods. The bullet player reference is then set to the player reference.

## Bullet Script

The bullet scripts main purpose is to add force using the update method so the bullet keeps moving then when the bullet collides with another object, the gravity direction is changed using:

PlayerReference.SetTargetGravityDirection(

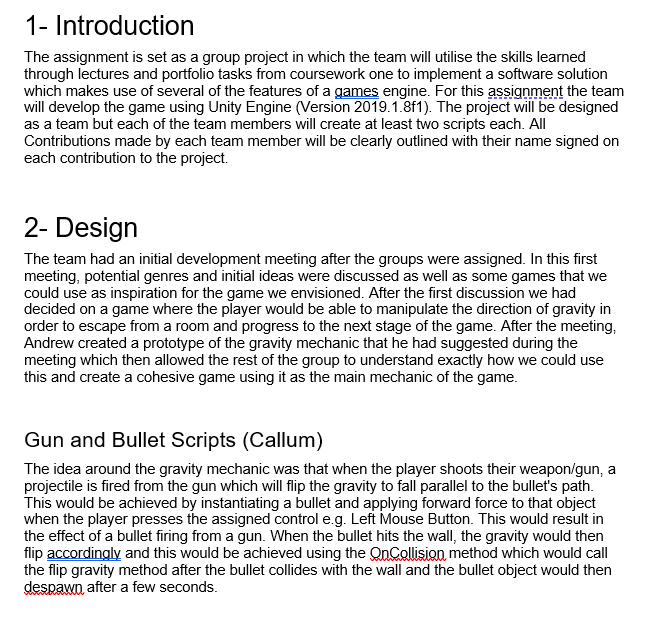
CustomMathLibrary.CalculateImpactNormalFromCollision(collision)

);

After the gravity is changed the bullet is then destroyed. The start method also handles the despawnBulletTime and the rBody variables.

## Group Report Contributions

Within the group report, we decided that each of us would contribute parts to the report in order to complete it to a high standard. I was assigned the introduction as well as writing a bit about the design of the game from our first few weeks and meetings as a team. I also wrote a small section in regard to my gun and bullet scripts which were my main contribution to the game. Below I have attached a screenshot of my work that I contributed to the group report.



## Testing Gun Script

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test no | Test description | Input | Expected outcome | Actual outcome | Pass/Fail |
| 1 | Check that gun works as intended when user clicks fire button | Mouse0/Left Mouse Button | Bullet is instantiated and travels forward from the gun | As expected | Pass |
| 2 | Check that the gun prefab is displayed correctly | N/A | Gun model asset should be loaded and shown on screen | As expected | Pass |
| 3 | Check that the gun fire-rate is working as intended | Mouse0/Left Mouse Button | User should not be able to fire as fast as they can click, limited to a few shots per second. | As expected | Pass |

## Testing Bullet Script

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test no | Test description | Input | Expected output | Actual output | Pass/Fail |
| 1 | Check that the bullet prefab is displayed correctly upon instantiation | Mouse0/Left Mouse Button | Bullet prefab can be seen fired from the end of the gun and travels forward until it collides with a wall and despawns after 3 seconds | As expected | Pass |
| 2 | Check that the bullet travels forward when instantiated | N/A | Bullet travels forward until it collides with an object, then falls to the floor | As expected | Pass |
| 3 | Check that the bullet object is destroyed, 3 seconds after colliding with another object. | N/A | Gravity is flipped to act down upon the wall the bullet hit | As expected | Pass |
| 4 | Check that the gravity if flipped when the bullet collides with the wall/object. |  |  | As expected | Pass |

# **Code Listings**

## Gun Script

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class GunScript : MonoBehaviour

{

//public float moveForce = 0f;

private Rigidbody rbody;

public GameObject bullet;

public Transform gun;

public float fireRate = 0f;

// public float bulletForce = 0f;

private float fireRateTimeStamp = 0f;

public CustomPlayerCharacter PlayerReference;

// Start is called before the first frame update

void Start()

{

rbody = GetComponent<Rigidbody>();

//PlayerReference = GetComponentInParent<CustomPlayerCharacter>();

}

// Update is called once per frame

void Update()

{

//float h = Input.GetAxisRaw("Horizontal") \* moveForce;

//float v = Input.GetAxisRaw("Vetical") \* moveForce;

//rbody.velocity = new Vector3(h, v, 0);

if (Input.GetKey(KeyCode.Mouse0))

{

if (Time.time > fireRateTimeStamp)

{

//FireGun();

// Add force to make bullet move forward //

fireRateTimeStamp = Time.time + fireRate;

}

}

}

public void FireGun()

{

// Instantiate bullet at guns position and rotation //

GameObject go = (GameObject)Instantiate(

bullet, this.transform.position + (this.transform.up \* 0.5f), this.transform.rotation);

go.GetComponent<BulletScript>().PlayerReference = PlayerReference;

// bullet player reference = player reference sotred in this class //

}

}

## Bullet Script

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class BulletScript : MonoBehaviour

{

public float despawnBulletTime = 0f;

//public Transform gun;

public float bulletForce = 10000f;

private Rigidbody rbody;

public CustomPlayerCharacter PlayerReference;

// Start is called before the first frame update

void Start()

{

Destroy(gameObject, despawnBulletTime);

rbody = GetComponent<Rigidbody>();

//PlayerReference = GetComponentInParent<CustomPlayerCharacter>();

}

// Update is called once per frame

void Update()

{

GetComponent<Rigidbody>().transform.position += rbody.transform.up \* bulletForce;//.AddForce(rbody.transform.forward \* bulletForce, ForceMode.Impulse);

}

private void OnCollisionEnter(Collision collision)

{

PlayerReference.SetTargetGravityDirection(

CustomMathLibrary.CalculateImpactNormalFromCollision(collision)

);

Destroy(this.gameObject);

}

}

# **References**

Firing Projectiles | Unity Quick Tutorial, Abhinav a.k.a Demkeys, March 20, 2016

<https://www.youtube.com/watch?v=L2lldjxHJTo>

Fire Guns in Unity - Unity Firing Bullets Tutorial | How to Fire Projectiles in Unity, Omnirift, August 7, 2017

<https://www.youtube.com/watch?v=1UekWA1osNw>

Unity FPS Course – Shooting Bullets, Zenva, March 12, 2017

<https://www.youtube.com/watch?v=6eIVxyxoimc>

Unity Shooting Tutorial (Raycast and Prefab), Brackeys, April 10, 2013

<https://www.youtube.com/watch?v=mpxim8YbsMk>