

# **Zig Zag Joy Game Document**

**Andhra Pradesh State Skill Development Corporation  
(APSSDC)**



**Collaboration with Kajaani University of Applied Sciences**



**PROUDLY PRESENTS**

**Indian Game Development Challenge-2018**



**With the help of**  
**SRM University, Amaravati, AP**



**Powered by**  
**Government of Andhra Pradesh**



**In association with**  
**Andhra Pradesh Economic Development Board**



**Document Designed**

**By**

**YANAMANDRA SRINIVASA SANDEEP KUMAR SARMA**

**Producer**

**From**

**Frnd Of Frnd's  
Studios**

# Contents

## 1.Team logo

- 1.1.Game logo

## ❖ 2.Game concept

- 2.1.Game overview

- ◆ 2.1.1.Game Genre

- 2.1.2.Game view

- ◆ 2.1.3.Game theme

- ◆ 2.1.4.Game play

## ❖ 3.Player Controls

- 3.1.Look and Feel of the game

- ◆ 3.2.Different levels of the game

- 3.2.1. Level transition Scene

## ❖ 4.Assets in the game

## ❖ 5.Character

- 5.1.Character design

## ❖ 6.Mechanics

## ❖ 7.Application Development

- 7.1. .Engine Used
  - ◆ 7.2. Art tools Used
    - 7.3. User interface

## ❖ 8.Code Snippets

- 8.1. Character Controller
  - 8.2. Collectible
    - ◆ 8.3. Health
    - ◆ 8.4. Destroy
    - ◆ 8.5. Move to position
    - ◆ 8.6. Animation
    - ◆ 8.7. Platform
    - ◆ 8.8. Score

## ❖ 9. Frnd of Frnd's Team Members

## 1.Team Logo



### 1.1.Game Logo



## **2. Game Concept: -**

The game is about catching the fruit and escape from the different Obstacles coming towards the player and to score points as much as possible.

### **2.1. Game Overview: -**

2.1.1 Game Genre: It is level based running game

2.1.2 Game view: Third person

2.1.3 Game Theme: The back drop of the game is set to with houses and trees.

2.1.4 Gameplay: When game starts our player has to select the a fruit which the player likes in a given couple of fruits. its an level based run game, player have to escape from the obstacles coming towards the player in the front. simultaneously our player will move forward to catch the fruit which he selected. If player get selected wrong fruit then energy level get reduced, so to fulfill that criteria, player need to catch the energy packet immediately otherwise they get failed. Our player has to escape from the arrows. Our player need to maintain the energy levels till they get into next level successfully.

## **3. Player Controls:**

The player has common controls which are: W, A, S, D, for movements and arrows (up, down, left, right) also. To jump Spacebar is used.

### **3.1. Look and Feel of the game: -**

The objects in the game are ancient medieval period of India, the design's we use are more and more attractive it seems to be like Indian old houses. the path we use is zig zag way so that it gives more excitement to the player.

### **3.2. Different levels of the game: -**

The game has 2 levels, in first level the user have to catch fruits which is selected previously and escape from the obstacles, Fulfill the minimum fruits requirements and coming to the second level the environment are seems to be like earthquake and landslides and also complete change of seasons. When the player full fill the minimum fruits requirement which are mentioned then only it will possess to enter into the second level.



3.2.1. Level transition Scene: The level transition scene consists of 'n' no of houses and also consists of grass, trees and so.



**Figure 1: Top view of Level transition scene**



**Figure 2: Back view of Level transition scene**





**Figure 3: Side view of Level transition scene**

**Level 1:** In level 1 player has to catch the fruit which is selected previously, the fruits we provide that are apple, banana. The obstacles are sudden path break which means the player have to run continuously without any pauses, player should not run back side and fulfill the minimum fruits requirements. If player catches the selected fruit score get incremented 5 points if not score get decrement 5 points and parallel energy level also get decremented so to fulfill that criteria player need to catch energy pack. so that only the energy level gets increase.



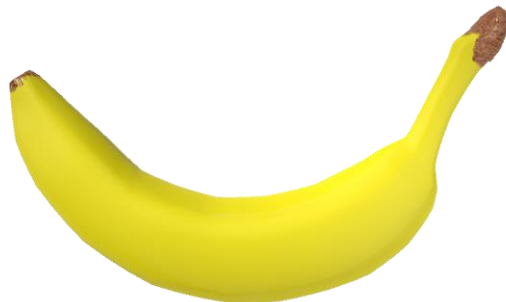
**Figure 4: The placement of fruits inside the level one in Level transition scene**

#### 4.Assets in the game: -

The game has different types of objects which are take able, in game there are intractable and non-intractable objects. the intractable objects are fruits, they are apple and banana the 3d model of the fruit objects are seems to be like.



**Figure 5: 3d model of apple fruit**



**Figure 6: 3d model of banana fruit**

The non-intractable objects are houses, grass, trees. when player unfortunately catches the wrong fruit his energy level get reduces so to fulfill that energy level he need to catch energy pack, the energy pack is designed in 3d model it seems to be like.



**Figure 7: Screenshot of energy pack**

If player catches this energy pack his energy level get increased gradually so player might be able to run, with that player will get continue his game play and perform running as much as possible.

The player might be able to see his energy level on the top left corner on game the energy level design as follows:



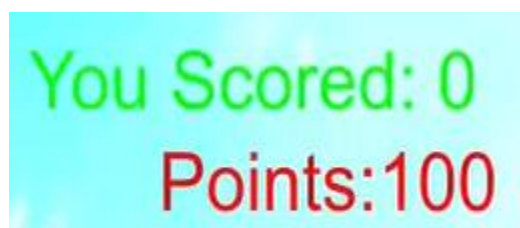
**Figure 8: Screenshot of energy level**

Player also get enable to his selected fruit on the top left corner on game the selected fruit text display's as follows:



**Figure 9: Screenshot of selected fruit**

Player can able to see how much he scored text as you scored message on the top right corner. and also the minimum fruits requirement named as points display's down of you scored, on the top right corner.



**Figure 10: Screenshot of right side notifications**

## 5.Character: -

The game has one character which is male character, the task of the character is to grab the fruits, run forward to chase the target and simultaneously escape from the obstacles. The character and other models are modelled, skinned and rigged in blender, fuse and are animated in Unity animator.the front view of character the as shown below:



**Figure 11: Screenshot of character in front view**

### 5.1.Character design: -

The character consist top innerwear and the outerwear with fully covered clothes. The character seems to be like the olden culture and suitable for that environment.



**Figure 12: Screenshot of character in side view**



**Figure 13: Screenshot of character in side view**



## 6.Mechanics: -

Some of the main mechanics in the game are:

**Picking up fruits:** In the game the player can pick up certain fruits by “press front arrow button”. Player can move to left and right directions, rotate by “press left & right arrow button”. And character can jump by “press space bar button” The scenery of character motions as follows:



**Figure 14: Screenshot of character moving front by pressing front arrow**



**Figure 15: Screenshot of character gets jump by pressing space bar**



## 7.Application Development: -

7.1. .Engine Used: The game is developed using “Unity” and C# script



Image: Unity logo source: [unity3d.com](https://unity3d.com)

7.2. Art tools Used: blender is used to houses, texture and the assets in this game. And also adobe fuse is used for character design.



Image: blender logo source: [blender.org](https://blender.org)



Image: fuse logo source: [BrandEPS.com](https://BrandEPS.com)

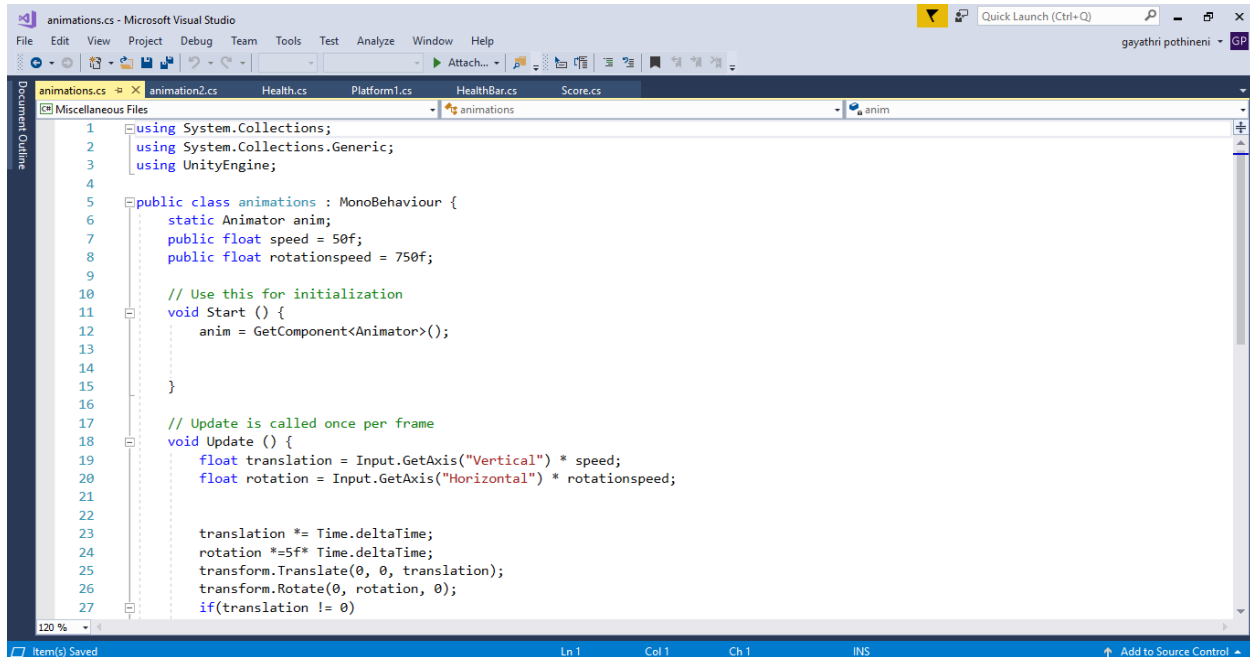
7.3. User interface: game logo, team logo and also menu design are developed by using photoshop tool software.



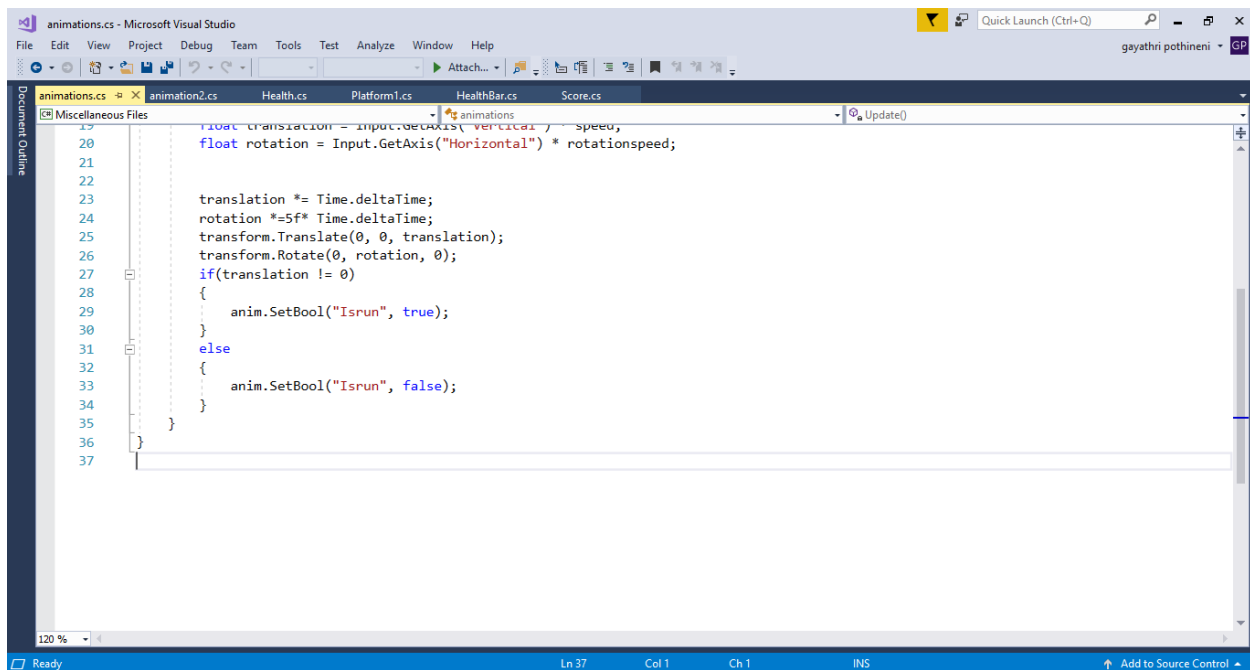
Image: photoshop logo source: [BrandEPS.com](https://BrandEPS.com)

## 8.Code Snippets: -

### 8.1. Character Controller:

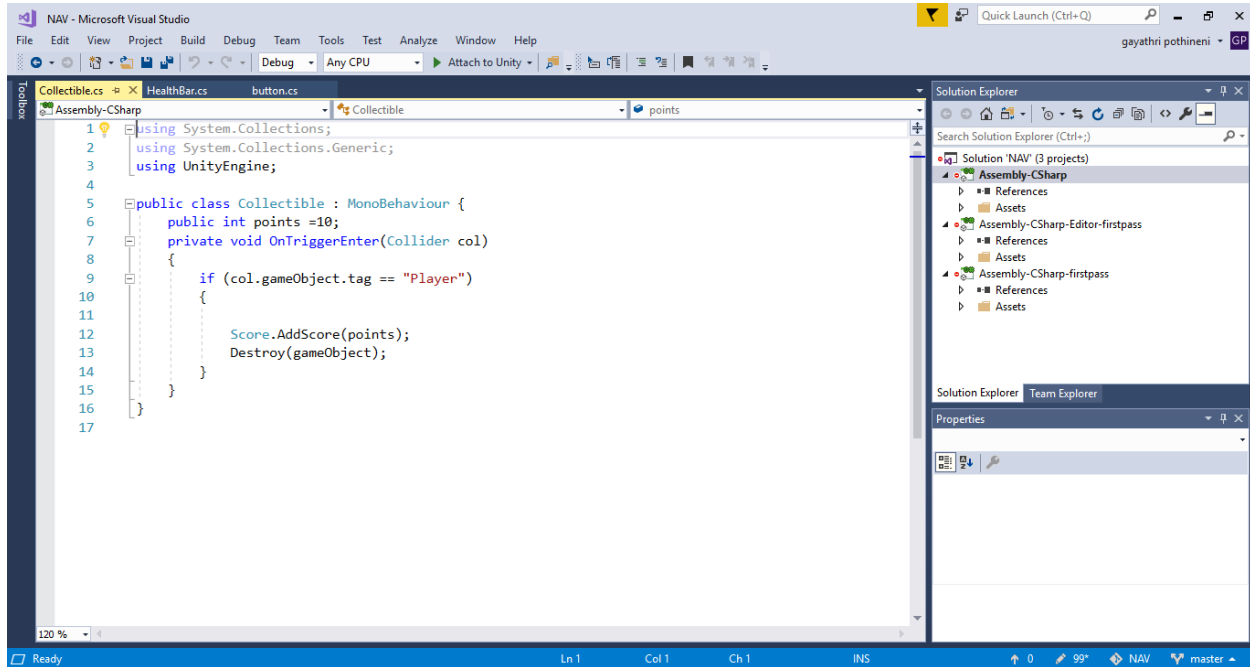


```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class animations : MonoBehaviour {
6     static Animator anim;
7     public float speed = 50f;
8     public float rotationspeed = 750f;
9
10    // Use this for initialization
11    void Start () {
12        anim = GetComponent<Animator>();
13    }
14
15    // Update is called once per frame
16    void Update () {
17        float translation = Input.GetAxis("Vertical") * speed;
18        float rotation = Input.GetAxis("Horizontal") * rotationspeed;
19
20        translation *= Time.deltaTime;
21        rotation *= 5f * Time.deltaTime;
22        transform.Translate(0, 0, translation);
23        transform.Rotate(0, rotation, 0);
24        if(translation != 0)
25    }
```

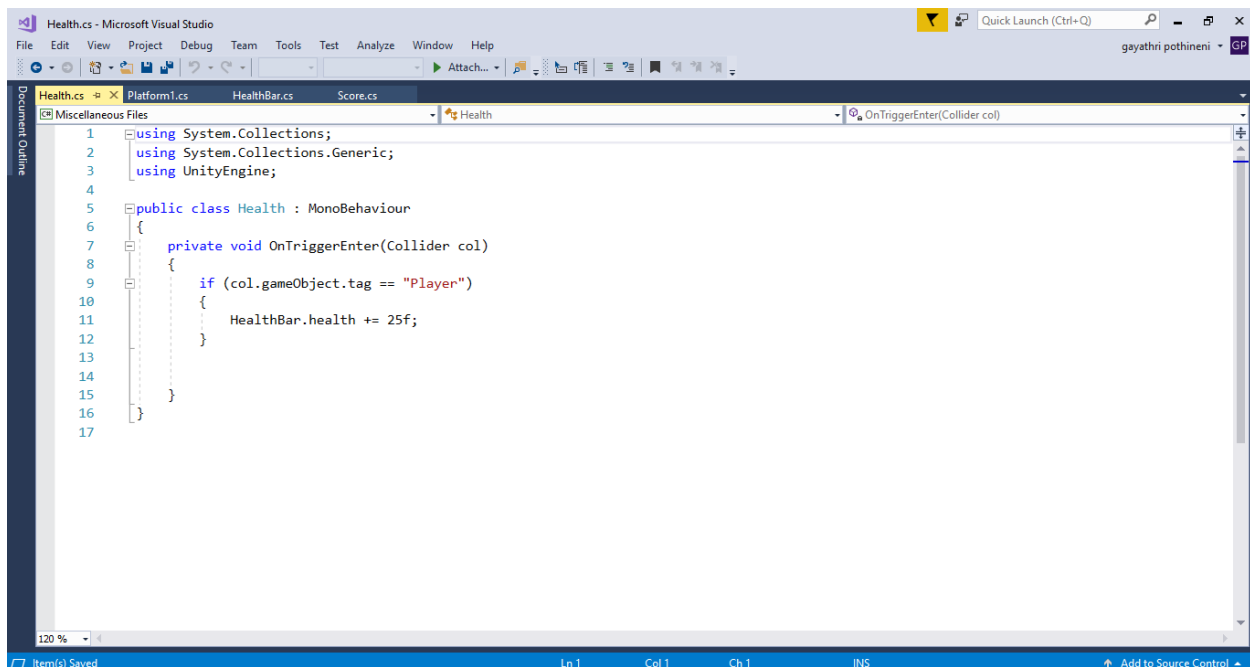


```
19 float translation = Input.GetAxis("Vertical") * speed;
20 float rotation = Input.GetAxis("Horizontal") * rotationspeed;
21
22    translation *= Time.deltaTime;
23    rotation *= 5f * Time.deltaTime;
24    transform.Translate(0, 0, translation);
25    transform.Rotate(0, rotation, 0);
26    if(translation != 0)
27    {
28        anim.SetBool("Isrun", true);
29    }
30    else
31    {
32        anim.SetBool("Isrun", false);
33    }
34 }
35
36
37 }
```

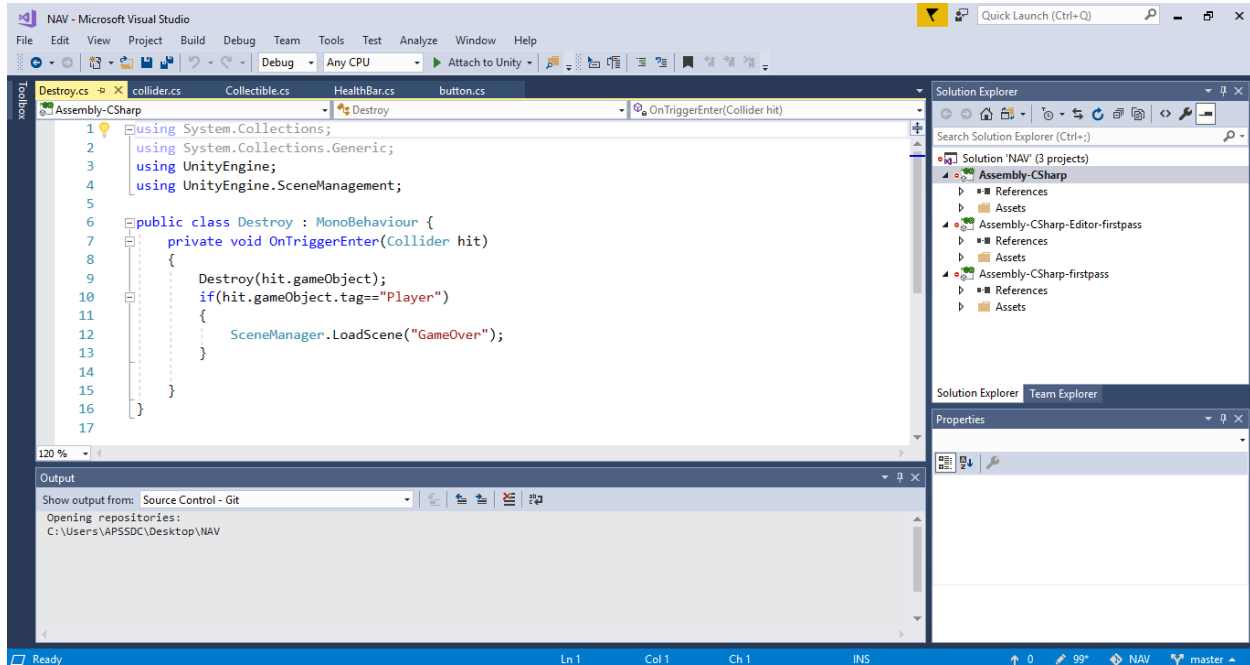
## 8.2. Collectible:



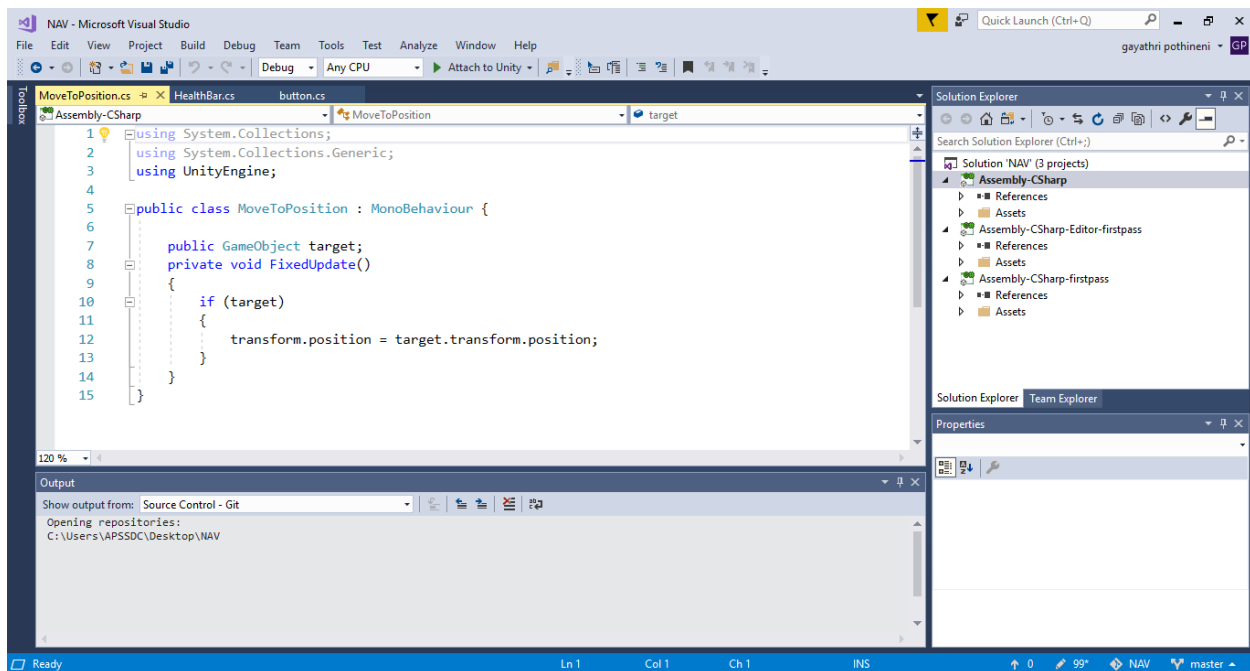
## 8.3. Health:



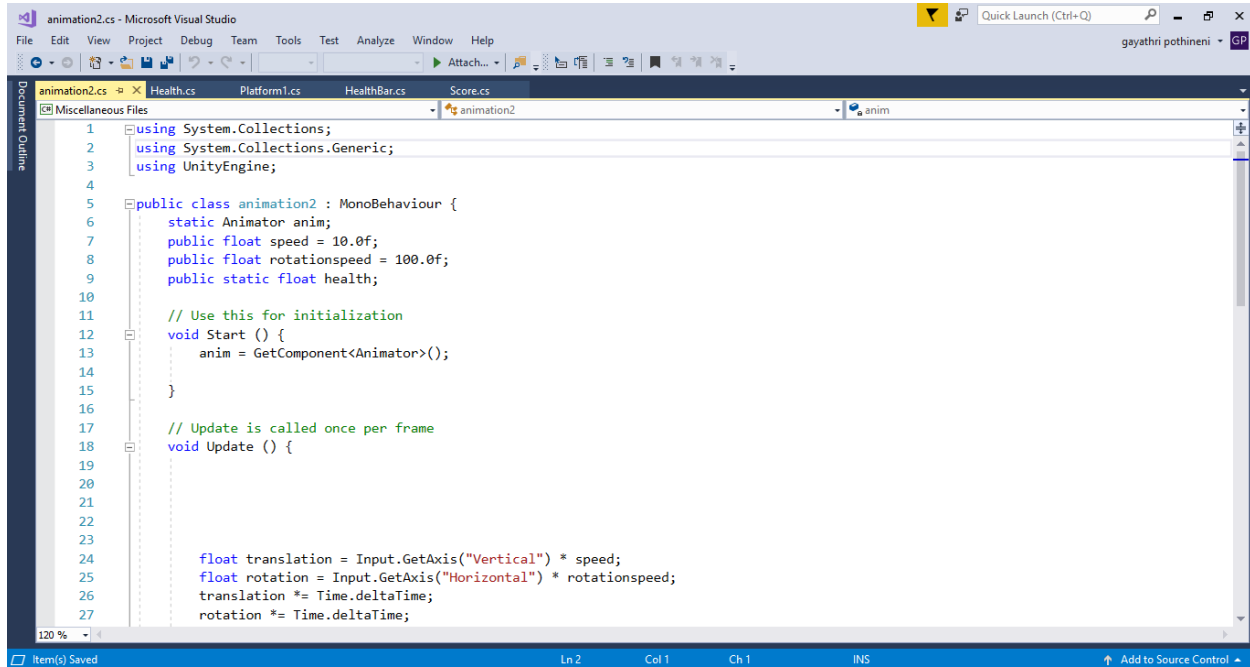
## 8.4. Destroy:



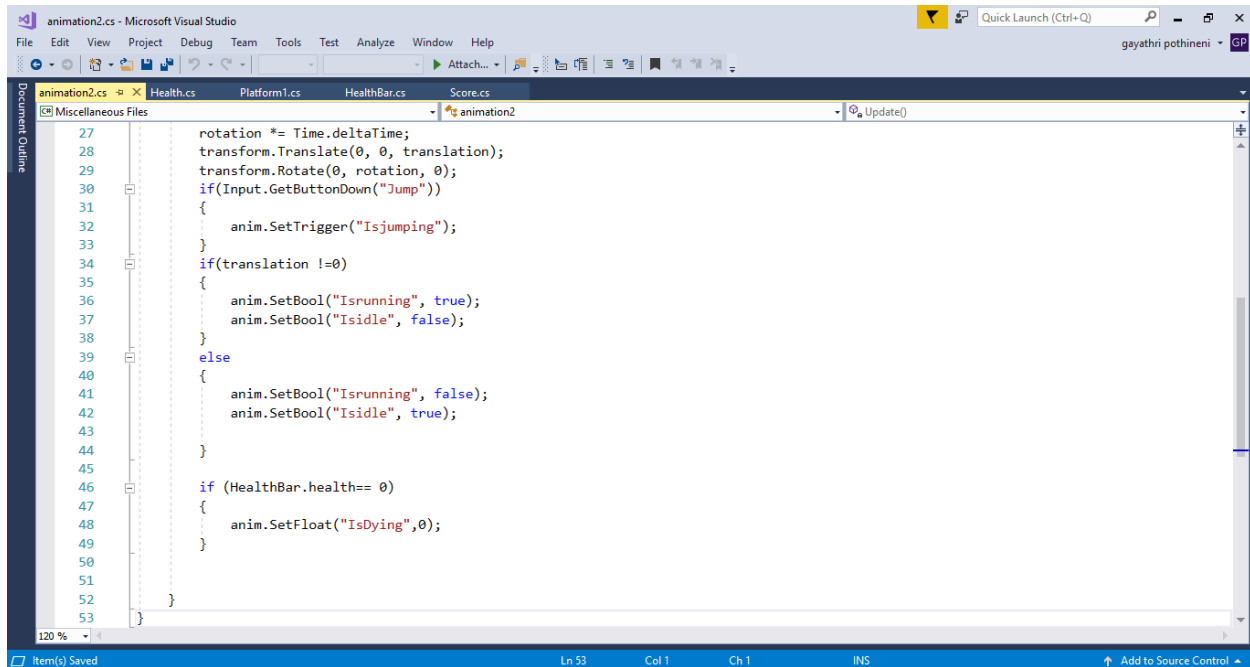
## 8.5. Move to position:



## 8.6. Animation:

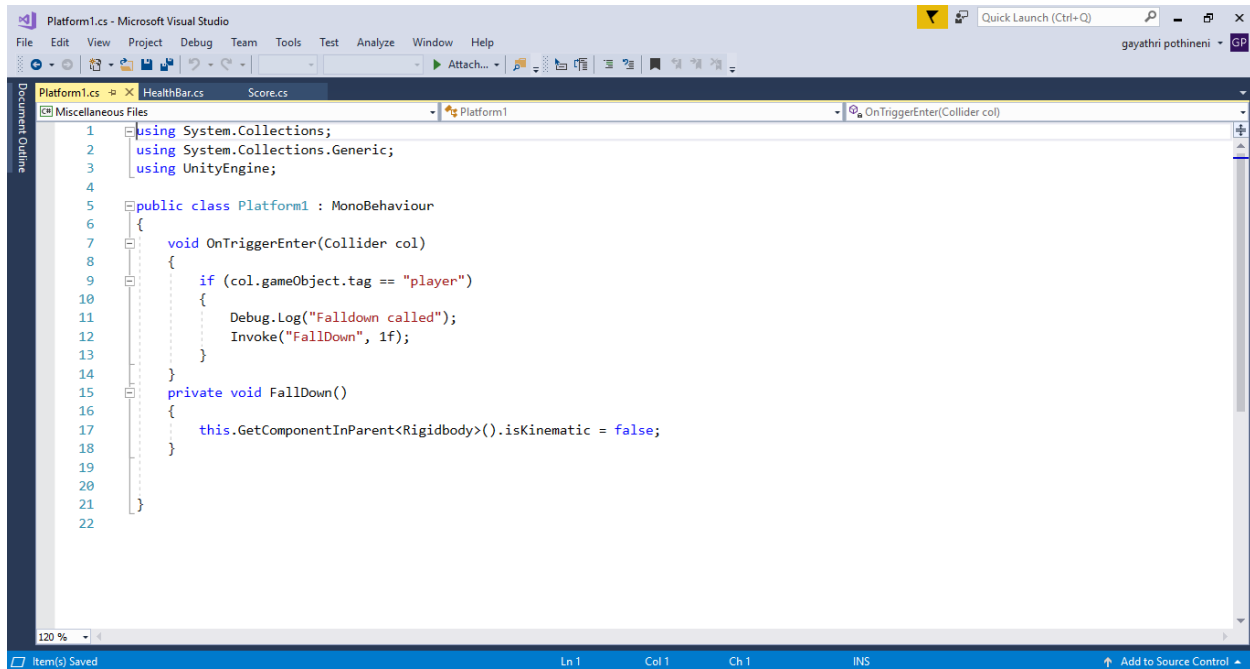


```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class animation2 : MonoBehaviour {
6     static Animator anim;
7     public float speed = 10.0f;
8     public float rotationspeed = 100.0f;
9     public static float health;
10
11     // Use this for initialization
12     void Start () {
13         anim = GetComponent<Animator>();
14     }
15
16
17     // Update is called once per frame
18     void Update () {
19
20
21
22
23
24         float translation = Input.GetAxis("Vertical") * speed;
25         float rotation = Input.GetAxis("Horizontal") * rotationspeed;
26         translation *= Time.deltaTime;
27         rotation *= Time.deltaTime;
```



```
27         rotation *= Time.deltaTime;
28         transform.Translate(0, 0, translation);
29         transform.Rotate(0, rotation, 0);
30         if(Input.GetButtonDown("Jump"))
31         {
32             anim.SetTrigger("Isjumping");
33         }
34         if(translation !=0)
35         {
36             anim.SetBool("Isrunning", true);
37             anim.SetBool("Isidle", false);
38         }
39         else
40         {
41             anim.SetBool("Isrunning", false);
42             anim.SetBool("Isidle", true);
43         }
44
45         if (HealthBar.health== 0)
46         {
47             anim.SetFloat("IsDying",0);
48         }
49
50
51     }
52
53 }
```

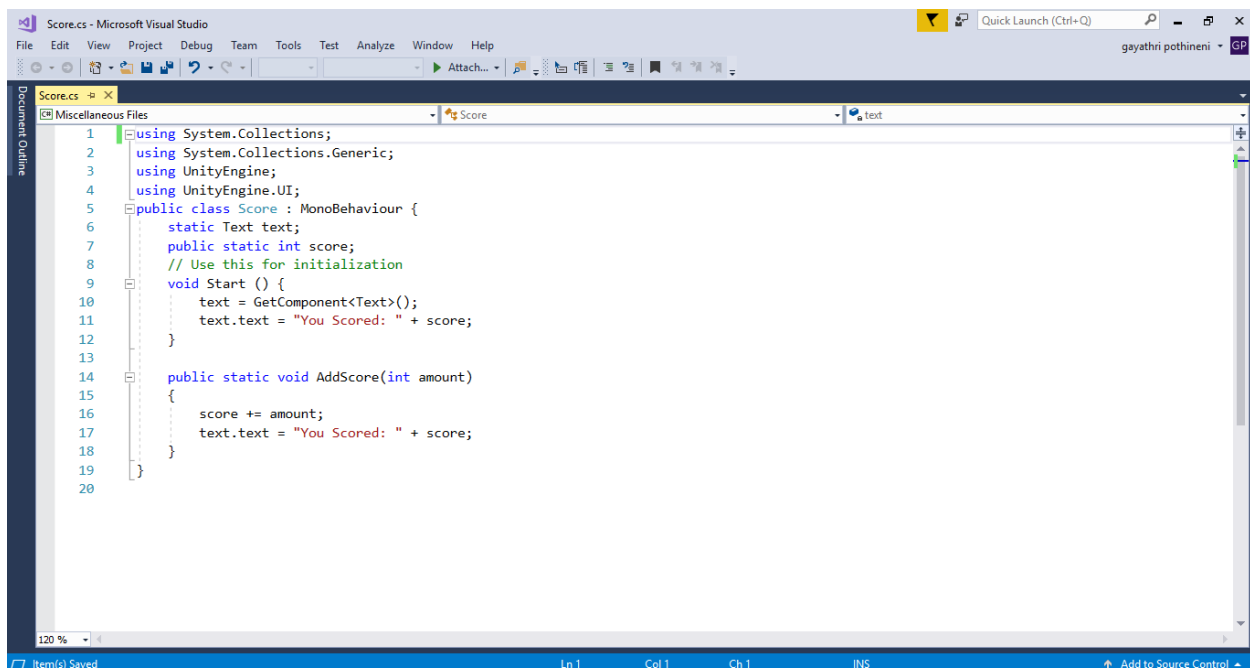
## 8.7. Platform:



The screenshot shows the Microsoft Visual Studio IDE with the Platform1.cs script open. The script is a C# class that inherits from MonoBehaviour. It contains two methods: OnTriggerEnter and FallDown. The OnTriggerEnter method checks if the collider's tag is "player". If so, it logs a message and calls the FallDown method. The FallDown method sets the isKinematic property of the rigidbody to false.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class Platform1 : MonoBehaviour
6 {
7     void OnTriggerEnter(Collider col)
8     {
9         if (col.gameObject.tag == "player")
10        {
11            Debug.Log("Falldown called");
12            Invoke("FallDown", 1f);
13        }
14    }
15    private void FallDown()
16    {
17        this.GetComponentInParent<Rigidbody>().isKinematic = false;
18    }
19 }
20
21
22
```

## 8.8. Score:



The screenshot shows the Microsoft Visual Studio IDE with the Score.cs script open. The script is a C# class that inherits from MonoBehaviour. It contains static variables for Text and int score. It has two methods: Start and AddScore. The Start method gets the Text component and sets its text to "You Scored: " + score. The AddScore method increments the score by the amount and updates the text.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5 public class Score : MonoBehaviour {
6     static Text text;
7     public static int score;
8     // Use this for initialization
9     void Start () {
10        text = GetComponent<Text>();
11        text.text = "You Scored: " + score;
12    }
13
14    public static void AddScore(int amount)
15    {
16        score += amount;
17        text.text = "You Scored: " + score;
18    }
19 }
20
```

## 9. Frnd of Frnd's Team Members

**PRODUCER**

*Y.S.Sandeep Kumar Sarma*

**PROGRAMER**

*B.Sahithi*

*P.Prasanna*

*K.Siva Lakshmi*

*M.Venkat Naveen*

*P.Gayathri Sowjanya*

**DESIGNER**

*K.Anilkumar*

*Y.Rajasekhar Reddy*

**ARTIST**

*K.Vamsi Krishna*

*D.Daneswari*