Game Programming Project 2 – Dude Can You Reach The Parachute?

Rules of the Game:
There will be a player which can be moved in directions LEFT and RIGHT and can be made to jump by clicking the SPACE bar or the UP arrow key.
There will be coins and bombs in the game.
The player has to make sure to avoid all the bombs or else the level will restart and he will have to play that particular level again.
The player has to collect as many coins as he can so that his score increases.
The player's main motive is to reach the parachute by collecting maximum number of coins and avoiding all the bombs which will come in his way.
Once he reaches the parachute, he will be transferred to the next level immediately and there are a total of three levels which he has to complete to Win the game.

I have the following scenes in my project:

1. Main Menu
2. Level 1
3. Level 2
4. Level 3
5. Win

I will be explaining each one in detail in this document.

Main Menu:

First, I had to create an empty GameObject in the Main Camera. I then changed that empty GameObject to UI – Canvas.
Inside the Canvas, I created a TextMeshPro – Text and formatted it a little bit according to the needs of the game.
I created a button on Play text ad well as Quit text.
When the Play Button is clicked the MainMenu Script is called and through that script we would move to the next scene present.
When the Quit Button is clicked the Application.Quit() is called which is used to quit the game.

Level 1:

The following GameObjects are there in Level 1:

1. CubeLevel1
2. Directional Light
3. SpawnManager
4. DeathTrigger
5. Backgroud
6. SoundManager
7. Canvas

CubeLevel1:
First, we added Box Collider 2D to this object and also added Rigidbody 2D.
This is the first platform we are creating so we have given it a location so as where to spawn.
The other platforms will be spawned dynamically by taking the location of this CubeLevel1 and for dynamic spawning we will be creating a Prefab of this and call it GroundLevel1.
We have added the script CoinSpawner to this object so that it can spawn coins automatically over this ground over the three locations which we have given to the system which are Coin1, Coin2 and Coin3.

```
using UnityEngine;
using System.Collections;

public class CoinSpawner : MonoBehaviour
{

    public Transform[] coinSpawns;
    public GameObject coin;

    // Use this for initialization
    void Start()
    {

        Spawn();
    }

    void Spawn()
    {
        for (int i = 0; i < coinSpawns.Length; i++)
        {
            int coinFlip = Random.Range(0, 2);
            if (coinFlip > 0)
                Instantiate(coin, coinSpawns[i].position, Quaternion.identity);
        }
    }

}
```

In this the coinSpawns Transform has the location of where the coins have to be spawned and GameObject coin is the prefab.
When the Spawn() method is called a for loop is called in which we use the Instantiate the function to spawn the coins on the three locations which we defined.

Directional Light:
The directional light is a GameObject which we created so that there is a little light on the main player.

SpawnManager:
The main function of this GameObject is to spawn grounds dynamically according to the location of the previous grounds spawned.
It is a script and is explained as follows:

```
using UnityEngine;
using System.Collections;

public class SpawnManagerWin : MonoBehaviour
{
```

```
    public int maxPlatforms = 20;
    public GameObject platform;
    public float horizontalMin = 7.5f;
    public float horizontalMax = 14f;
    public float verticalMin = -6f;
    public float verticalMax = 6;
    public GameObject parachute;


    private Vector2 originPosition;


    void Start()
    {

        originPosition = transform.position;
        Spawn();

    }

    void Spawn()
    {
        for (int i = 0; i < maxPlatforms; i++)
        {
            Vector2 randomPosition = originPosition + new Vector2(Random.Range(horizon
talMin, horizontalMax), Random.Range(verticalMin, verticalMax));
            Instantiate(platform, randomPosition, Quaternion.identity);
            originPosition = randomPosition;
        }
        Instantiate(parachute, originPosition, Quaternion.identity);
    }

}
```

We have defined few variables in the start. When the spawn method is called, first thing it does is it goes in the for loop and checks if already maximum number of platforms are spawned then it does not spawn another platform, it just goes out of the for loop and spawns the parachute as seen in the above program by using the Instantiate function. Inside, the for loop the Vector2 randomPosition is used to spawn a new platform at random position using the function Random.range as seen in the above program.

DeathTrigger:
This GameObject is another rectangle which we have created and we have added the component Box Collider 2D and we have made the Is Trigger to on. In this we have also added the script DeathTrigger which is explained as follows:

```
using UnityEngine;
using System.Collections;

public class DeathTrigger : MonoBehaviour
{

    // Use this for initialization
    void Start()
    {

    }

    // Update is called once per frame
```

```
    void Update()
    {

    }

    void OnTriggerEnter2D(Collider2D other)
    {
        if (other.gameObject.CompareTag("Player"))
            Application.LoadLevel(Application.loadedLevel);
    }
}
```

The main function of this script is to load the current level again ie when the Player collides with the DeathTrigger then Application.LoadLevel(Application.loadedLevel); is called and the level is restarted.

Background:

This is a sound GameObject. The main function of this GameObject is to play the background sound. The only component we added in this was the Audio Source and to this component we gave the Audio file.

SoundManager:

This is also a GameObject which only has the script as its component. The script is explained as follows:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class SoundManagerScript : MonoBehaviour
{
    // Start is called before the first frame update
    public static AudioClip coinSound;
    static AudioSource audioSource;

    void Start()
    {

        coinSound = Resources.Load<AudioClip>("coin");

        audioSource = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void Update()
    {

    }

    public static void PlaySound(string clip)
    {
        switch (clip)
        {

            case "coin":
                audioSource.PlayOneShot(coinSound);
                break;

        }
```

```
    }
}
```

When this function is called then it first loads the sound which we have given. Then when the PlaySound method is called then it will play the sound for which the case matches once.

Canvas:
The canvas has a text Coins, which in turn has a script component attached to it and main function of it is to count the coins and display it on the Win scene. The script is as follows:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class CoinScript : MonoBehaviour
{
    public static int coinValue = 0;
    Text coins;

    // Start is called before the first frame update
    void Start()
    {
        coins = GetComponent<Text>();
    }

    // Update is called once per frame
    void Update()
    {
        coins.text = "Coins: " + coinValue;
    }


}
```

Level 2:

Most of the GameObjects are same as that of Level 1 but due to increasing difficulty, we have added bombs to this level and following is the bomb script:

```
using UnityEngine;
using System.Collections;

public class Bomb : MonoBehaviour
{

    // Use this for initialization
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }

    private void OnTriggerEnter2D(Collider2D other)
```

```
    {
        if (other.gameObject.CompareTag("Player"))
        {
            Application.LoadLevel(Application.loadedLevel);
        }
    }

}
```

The following is the BombSpawner Script:

```
using UnityEngine;
using System.Collections;

public class BombSpawner : MonoBehaviour
{

    public Transform[] bombSpawns;
    public GameObject bomb;

    // Use this for initialization
    void Awake()
    {

        Spawn();
    }

    void Spawn()
    {
        for (int i = 0; i < bombSpawns.Length; i++)
        {
            int bombFlip = Random.Range(0, 2);
            if (bombFlip > 0)
                Instantiate(bomb, bombSpawns[i].position, Quaternion.identity);
        }
    }

}
```

This script is same like the CoinSpawner script but instead of spawning coins its main function is to spawn bombs.

Level 3:

Most of the GameObjects are same as that of Level 1 but due to increasing difficulty, we have added bomb and also platform falls after the interval of 1 second.

The following is the PlatformFall Script:

```
using UnityEngine;
using System.Collections;

public class PlatformFall : MonoBehaviour
{

    public float fallDelay = 1f;


    private Rigidbody2D rb2d;

    void Awake()
    {
```

```
        rb2d = GetComponent<Rigidbody2D>();
    }

    void OnCollisionEnter2D(Collision2D other)
    {
        if (other.gameObject.CompareTag("Player"))
        {
            Invoke("Fall", fallDelay);
        }
    }

    void Fall()
    {
        rb2d.isKinematic = false;
    }



}
```

We also have the SimplePlatformController script in this Level which is used to check if the player is touching the ground for a certain amount of seconds then it makes the ground to fall down.

```
using UnityEngine;
using System.Collections;

public class SimplePlatformController : MonoBehaviour
{

    [HideInInspector] public bool facingRight = true;
    [HideInInspector] public bool jump = false;
    public float moveForce = 365f;
    public float maxSpeed = 5f;
    public float jumpForce = 1000f;
    public Transform groundCheck;


    private bool grounded = false;
    private Animator anim;
    private Rigidbody2D rb2d;


    // Use this for initialization
    void Awake()
    {
        anim = GetComponent<Animator>();
        rb2d = GetComponent<Rigidbody2D>();
    }

    // Update is called once per frame
    void Update()
    {
        grounded = Physics2D.Linecast(transform.position, groundCheck.position, 1 << L
ayerMask.NameToLayer("Ground"));

        if (Input.GetButtonDown("Jump") && grounded)
        {
            jump = true;
        }
    }
```

```
    void FixedUpdate()
    {
        float h = Input.GetAxis("Horizontal");

        anim.SetFloat("Speed", Mathf.Abs(h));

        if (h * rb2d.velocity.x < maxSpeed)
            rb2d.AddForce(Vector2.right * h * moveForce);

        if (Mathf.Abs(rb2d.velocity.x) > maxSpeed)
            rb2d.velocity = new Vector2(Mathf.Sign(rb2d.velocity.x) * maxSpeed, rb2d.v
elocity.y);

        if (h > 0 && !facingRight)
            Flip();
        else if (h < 0 && facingRight)
            Flip();

        if (jump)
        {
            anim.SetTrigger("Jump");
            rb2d.AddForce(new Vector2(0f, jumpForce));
            jump = false;
        }
    }


    void Flip()
    {
        facingRight = !facingRight;
        Vector3 theScale = transform.localScale;
        theScale.x *= -1;
        transform.localScale = theScale;
    }
}
```

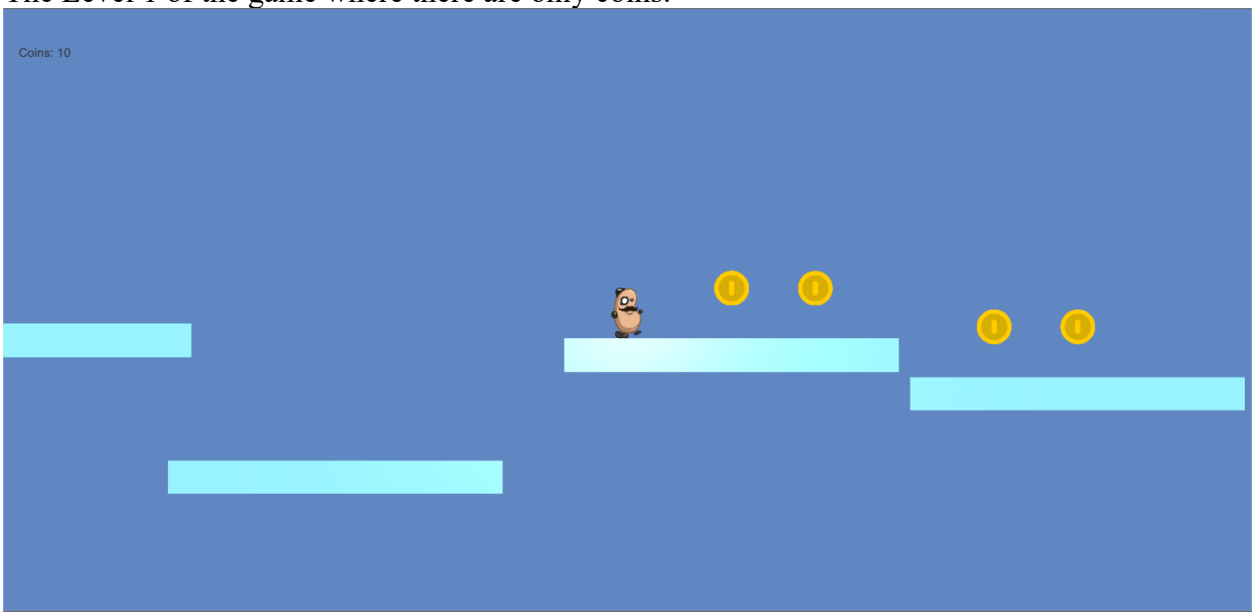
Win:

The Win scene has only TextMeshPro – Text which shows that you have won the game. It also shows, the total number of coins collected during the entire game. If you are able to reach this scene then you have successfully completed the game and won the game.

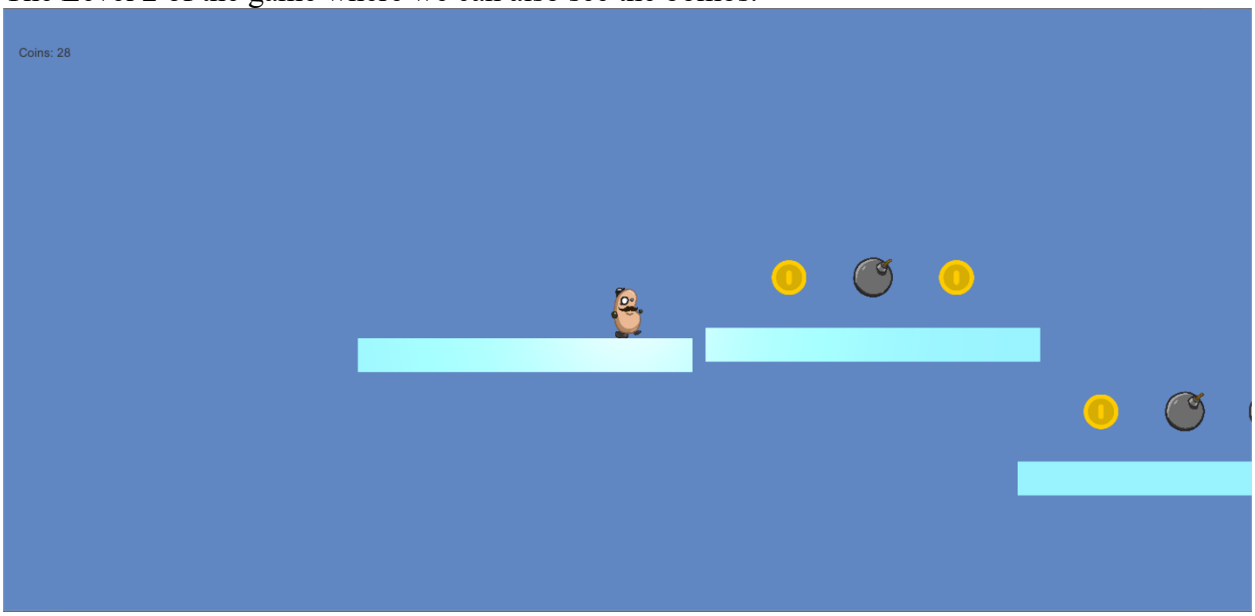
Screenshots of the Game:

The Main Menu of the Game:

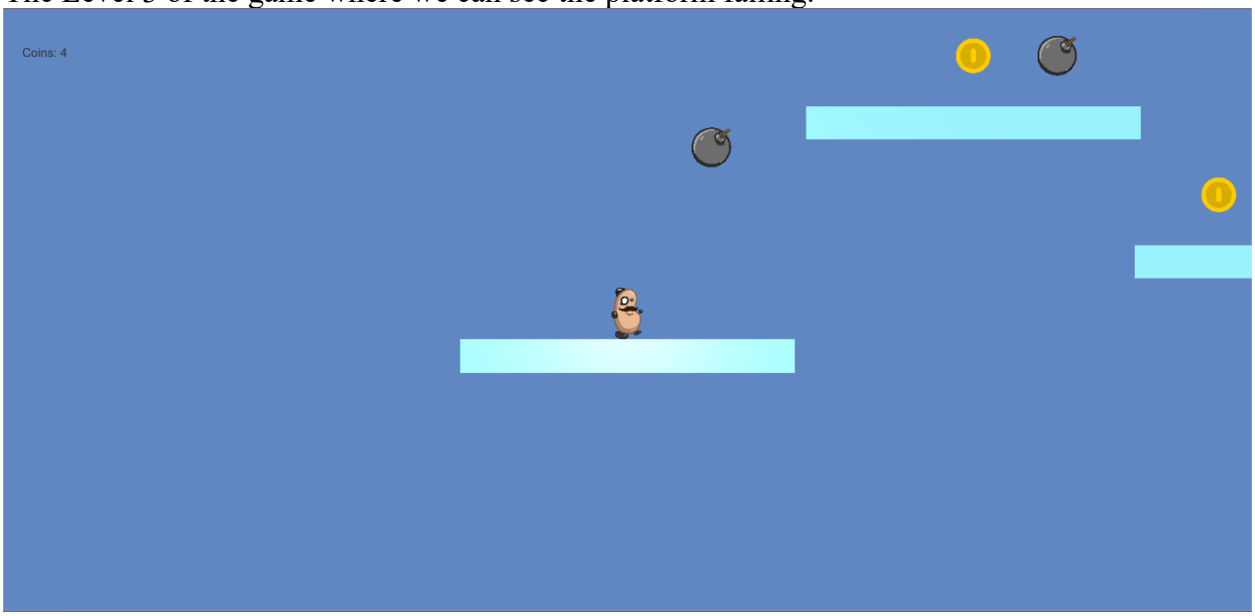
The Level 1 of the game where there are only coins.



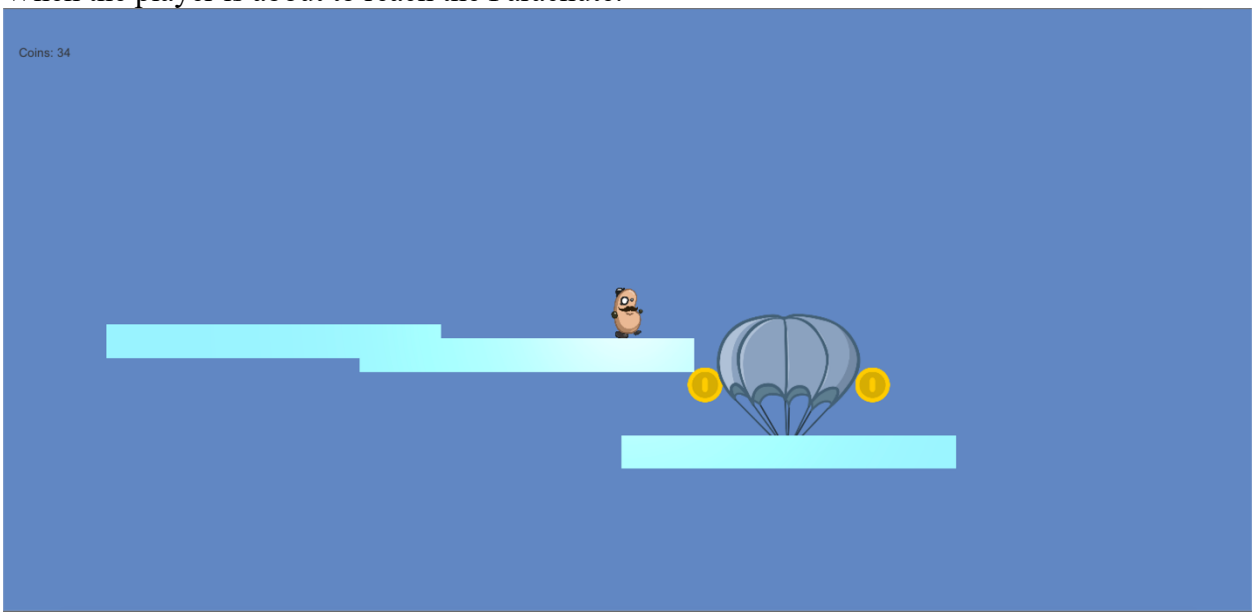
The Level 2 of the game where we can also see the bombs.



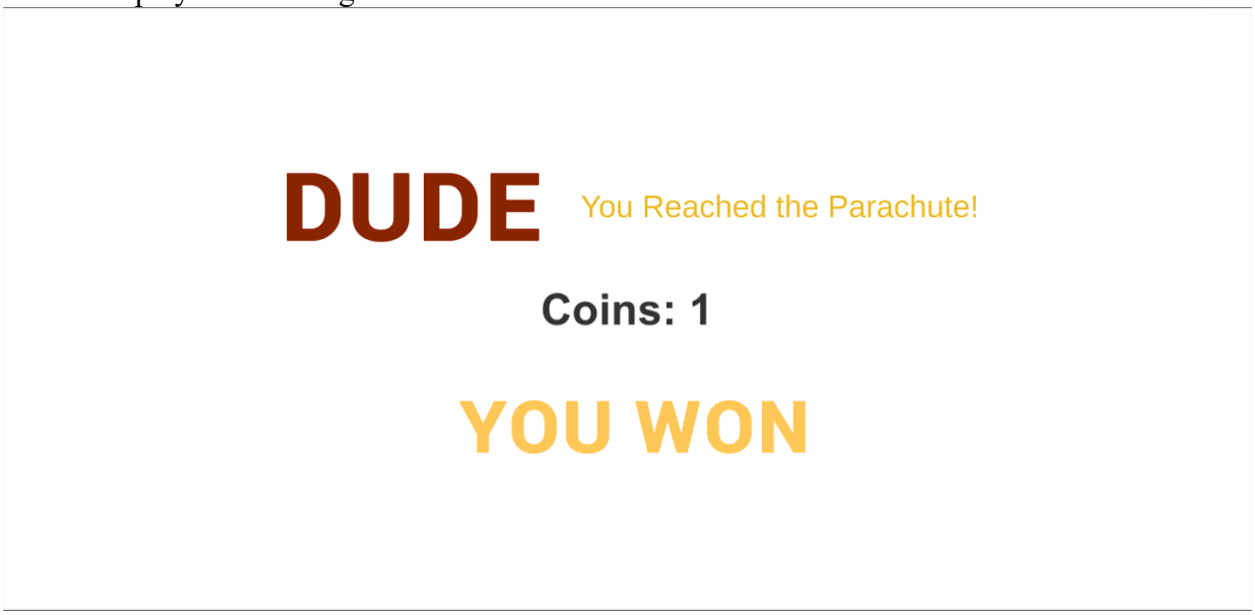
The Level 3 of the game where we can see the platform falling.



When the player is about to reach the Parachute.



When the player wins the game.



How To Play:

1. **Extract the zip file**
2. **Open Unity, Go to file Open Project then Select the Extracted Folder**
3. **Click on the Play Button in Unity**
4. **Play**

Submitted By:
Tejas Wadiwala
0889445