Duck Game(2D RPG Game)

Xianbin Wu
xianbin.wu001@umb.edu
University of Massachusetts Boston

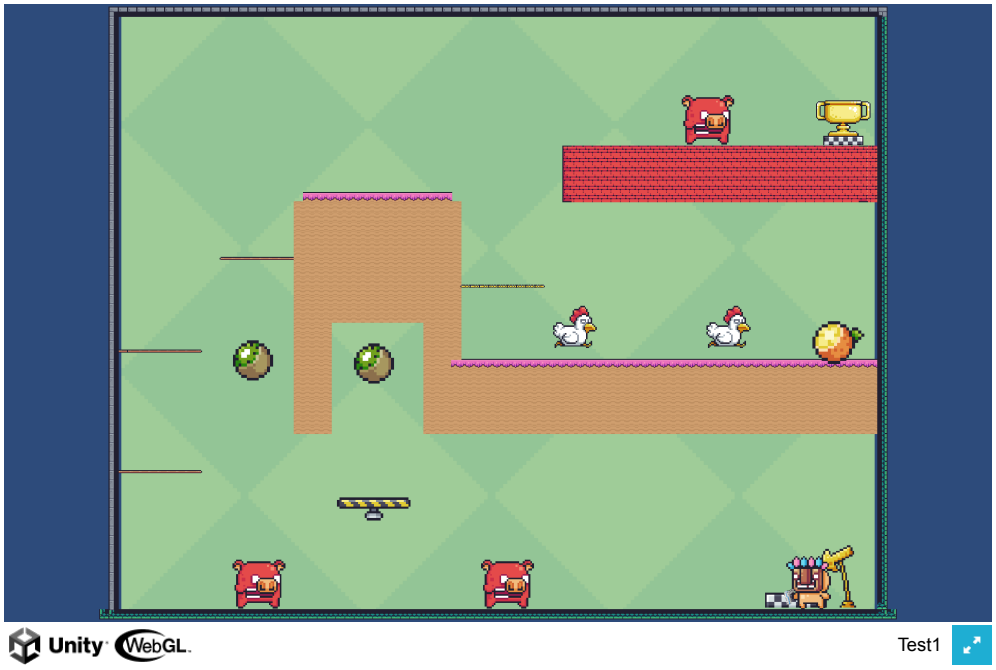


Figure 1: Stage 3 of the game

ABSTRACT

The WebGL Build option in Unity makes it possible for me to create a 2D role-playing game that is comparable to the Duck game that is available on Steam and makes it available for everyone to play on it. There are many different things that the player may do in this game. For example, they can attempt to avoid the enemy and reach the checkpoint, or they can try to harvest fruit on the side for fun. There are many other things that the player can do. Additionally, at the beginning of the game, you will be given a gun, with which you will have the ability to kill any animal that is in your path, whether it be a duck, chicken, pig, or bird. During this phase of the process, you will have the opportunity to make advantage of the player's animations, including jumping, wall jumping, double jumping, and firing a bullet out, in order to reach your target.

CS460, Fall 2022, Boston, MA

ACM ISBN 1337.
https://CS460.org

KEYWORDS

WebGL, Visualization, Unity, Animation,RPG

ACM Reference Format:
Xianbin Wu. 2022. Duck Game(2D RPG Game). In *CS460: Computer Graphics at UMass Boston, Fall 2022*. Boston, MA, USA, 4 pages. https://CS460.org

1 INTRODUCTION

My decision to work on this project was based on the fact that I had never before attempted to create a two-dimensional video game. Prior to making a decision regarding the project, I was pondering what sort of the end project will come to my mind. After playing Duck Game on Steam, I got an idea for how I might make a game of that kind. I conducted a lot of research online to figure out how to develop a game like this, and the program that came up most frequently as a recommendation was Unity. Unity WebGLPlatForm was chosen as the foundation for this game since I intend for it to be displayed on the website as well.

2 RELATED WORK

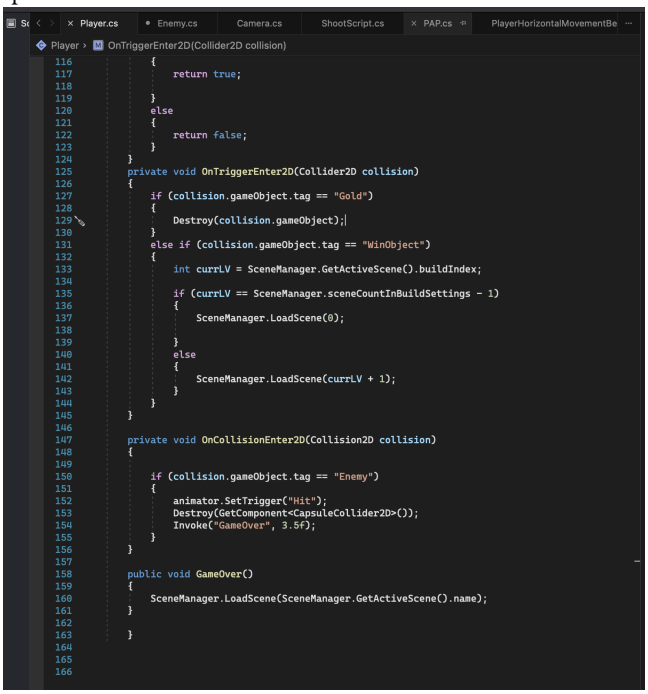
Here you can cite existing related work like Animator [**?**] or Camera [**?**], C C,GUI,HTML,Blender,Materials, Assignment 4

3 METHOD

In this game, you will undergo reanimation at the left tile on the first level of the game, and you will then have the opportunity to attempt to walk on the ground. There is friction on the ground, which will make it slightly more difficult to walk about. This will make it more difficult for players to navigate their way around. You will come across a chicken in the midst of the game at some point while you are moving around and experimenting with the jumping movement. You might try to dodge it or you could use the weapon that is currently in your hand to kill it. In order to use the weapon, you need to hit the left mouse button, and when you do that, you will see a bullet going out of there. If you want to kill the chicken, you need to give it at least three or more shots rather than just one. The chicken won't die from only one shot. After you have finished off the chicken, you will see that a single jump will not allow you to reach the middle platform where the bird was standing. In order to go to that location, you will need to utilize the animation for a double jump. You will also see fruit on the side of the path that you are walking to get to the flag, and you can pick some of that up along the route. Although it accomplishes nothing at the moment, you might want to pick it up just for fun. Also, if you have tried to jump to the wall, you will notice that there are functions of wall jump and wall slide motion when you are interacting with the wall. These functions are activated when you land on the wall. There will be three stages in the game, and you should do everything you can to stay alive until stage 3, as this will be the last stage of the game.

3.1 Implementation

The project would begin on the Unity WebGlPlatform, and, to the best of the developer's knowledge, they would use Unity both as a backup and to construct the game's foundation. This step involves using the Player Animation as well as using the manner of the animation of knowledge to determine which stage the player or the enemy should be at in order to remain idle, hit, jump, and so on. When I was playing the game, I spent a lot of time trying to figure out what the player was supposed to be doing at each particular point in the animation. In the meantime, a significant number of scripts were developed to govern the player's movement within the game, and as a result, the entirety of the game is based on those scripts.



```
Player.cs | Enemy.cs | Camera.cs | ShootScript.cs | PAP.cs | PlayerHorizontalMovementBe
Enemy › No selection
    private Animator animator;
    private int dir = 1;
    private float startPosX;
    private bool isHit = false;

    // Start is called before the first frame update
    void Start()
    {
        animator = GetComponent<Animator>();
        startPosX = transform.position.x;

    }

    // Update is called once per frame
    void Update()
    {
        if(isHit)
        {
            return;
        }
        transform.Translate(transform.right * Time.deltaTime * 0.4f * dir);

        if (transform.position.x > startPosX + 0.5)
            {
            dir = dir * -1;
            GetComponent<SpriteRenderer>().flipX = false;
        }
        else if(transform.position.x <startPosX - 0.5f)
        {
            dir = dir * -1;
            GetComponent<SpriteRenderer>().flipX = true;
        }
    }
    private void OntriggerEnter2D(Collider2D collision)
    {
        if(collision.gameObject.tag =="Player")
        {
            animator.SetTrigger("Hit");
            transform.position = new Vector3(transform.position.x, transform.position.y - 0.(
            isHit = true;
            BoxCollider2D[] collider2Ds;
            collider2Ds = GetComponents<BoxCollider2D>();
            foreach (var item in collider2Ds)
            {
                Destroy(item);
            }
            Destroy(GetComponent<Rigidbody2D>());
            Destroy(gameObject, 1F);
        }
    }
```

3.2 Milestones

3.2.1 Milestone 1. When I first started working on this game, I conducted a lot of research on it and checked out how each platform could function. I also made a decision regarding the type of game that should be developed. After thinking about all of the platforms and games that I want to develop, I have decided to focus on developing this game instead.

3.2.2 Milestone 2. I add the background for every stage of the game and created a tile map.

3.2.3 Milestone 3. I created the Player interface and animation for the player to jump, double jump, wall jump, and hit animation.

3.2.4 Milestone 4. I created the enemy scripts in order to stop the player to get to the Flag.

3.2.5 Milestone 5. I created the fruit to show some animation there as well.

3.2.6 Milestone 6. I finished all the scripts for the player and enemy in order for them to work properly.

3.3 Challenges

Describe the challenges you faced.

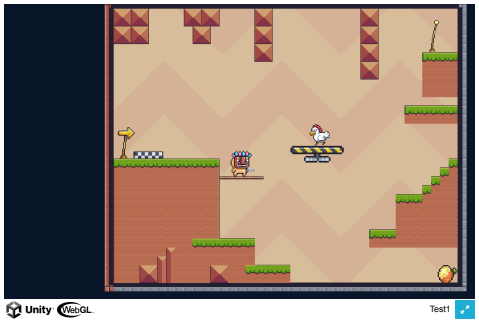
- Challenge 1: I misunderstood how to set a unit of tile, and take me long enough to start thinking about quitting designing the background. So, after that, I figure out a tile map that is way faster than filling each pixel by myself.
- Challenge 2: The player animation is way too much to handle at the first, I have to create every single script for each animation that is displayed in the game. It takes me almost 1/2 of the project time to do those things.
- Challenge 3: I have to organize every animation for every movement of the player in order to make an order of each in the timeline.

```
PlayerDoubleJumpBehaviour.cs | WallSlideBehaviour.cs | Damageable.cs | Hit.cs | Player.cs
Player › Update()
    {
        float moveX = Input.GetAxisRaw(PAP.axisXinput );

        animator.SetFloat(PAP.moveX, moveX);
        bool isMoving = Mathf.Approximately(moveX, 0f);

        animator.SetBool(PAP.isMoving, isMoving);

        bool lastOnGround = animator.GetBool(PAP.isOnGround);
        bool newOnGround = CheckIfOnGround();
        animator.SetBool(PAP.isOnGround, newOnGround);

        if(lastOnGround == false && newOnGround == true)
        {
            animator.SetTrigger(PAP.landedOnGround);
        }
        else
        {
            animator.ResetTrigger(PAP.landedOnGround);
        }

        bool onWall = CheckIfOnWall();
        animator.SetBool(PAP.isOnWall, onWall);

        bool isJumpKeyPressed = Input.GetButtonDown(PAP.jumpKeyName);

        if (isJumpKeyPressed)
        {
            animator.SetTrigger(PAP.JumpTriggerName);

        }

    }
    void FixedUpdate()
    {
        float forceX = animator.GetFloat(PAP.forceX);

        if (forceX != 0) rb.AddForce(new UnityEngine.Vector2(forceX, 0) * Time.deltaTime);

        float impulseY = animator.GetFloat(PAP.impulseY);
        float impulseX = animator.GetFloat(PAP.impulseX);

        if (impulseY != 0 || impulseX != 0)
        {
            float xDirectionSign = Mathf.Sign(transform.localScale.x);



            rb.AddForce(new UnityEngine.Vector2(xDirectionSign *impulseX,impulseY), ForceMoc
            animator.SetFloat(PAP.impulseY, 0);
            animator.SetFloat(PAP.impulseY, 0);
```

Figure 2: Player code

4 RESULTS

The finished product is satisfactory, but it has the potential to be even better. It is just a matter of time till every single minor movement and animation fault that it will be encountering is fixed. The game has turned out to be nearly what I believe it ought to be in the first place, despite the fact that it ought to have more content than that. The reason for this is that time is a bit too limited for me to manage every single component of the game, and I am working on it all by myself. Because of this, it does not turn out the way I believe it would, but I still consider it to be within the acceptable range.



5 CONCLUSIONS

This project served as a demonstration of a skill that I possess because it involved the creation of a video game through the application of Unity's technologies, specifically the Unity Scripting API, C, and the Unity WebGL Build, as well as the animation of the player and the adversary in the game. The project is still capable of being improved in a great number of different ways, and doing so in the future will require both a comprehension of the notion of front-end and back-end work as well as its implementation. I think that it will be really interesting as well as valuable for my future and the job that I will be doing in the future, and I enjoy the notion of doing this kind of final project as a capstone experience. I am not going to sugarcoat it; in the beginning, it was incredibly challenging because I was the sole creator and was responsible for everything. This kind of work has helped me become more aware of what I am capable of achieving within the allotted amount of time, and I believe that it should be the first step for anyone who is developing a 2D or 3D video game. In conclusion, I would want to express my gratitude to anyone who has read this entire article; the project that I have been working on has been rather intriguing.

REFERENCES

[1]Top Free Game Assets Tagged 2D." Itch.io, https://itch.io/game-assets/free/tag-2d. [2]Brackeys. "2D Animation in Unity (Tutorial)." YouTube, YouTube, 5 Aug. 2018, https://www.youtube.com/watch?v=hkaysu1Z-N8. [3]"How to Make 2D Shooting in Unity with Gun Recoil - Unity 2D." YouTube, YouTube, 21 Apr. 2020, https://www.youtube.com/watch?v=NKF-FkDzE-s. [4]darkskeletontuts. "Make a 2D Platformer Character in Unity - Complete Course for Beginners." YouTube, YouTube, 13 Sept. 2021, https://www.youtube.com/watch?v=aTB-5aE0POw.