GAZİ ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

CENG447 GAME DEVELOPMENT

Dönem Sonu Final Projesi

Dr. Murat YILMAZ

21118080409 - Bengi Kayra ÖZDEMİR
21118080800 - Emir DOĞAN

2024

Game Design Document

GAME's Name: Bortecine Run

Student Name(s): Emir Doğan, Bengi Kayra Özdemir

Introduction

In a time when the Turkish ancestors faced harsh Mongol and Chinese invasions, a female wolf discovers a baby abandoned in the mountains. Driven by maternal instincts, she saves the child and begins raising him. This child will grow up to become Oğuz Kağan, one of the most legendary heroes and leaders in Turkish mythology. Wanting to shield the baby from further violence, the wolf sets out to find a peaceful place where she can protect him. Recognizing the child's future potential, she also intends to train him in leadership and warrior skills. Their journey takes them through the perilous Iron Mountains, where they face numerous challenges. With no water and a long, treacherous path ahead, the wolf must move swiftly, knowing that any obstacle could be fatal, and the future hero could be lost, along with Turkey's destiny.

Problem

The main objective of the game is for the player to take on the role of the wolf mother, navigating through a straight road while carrying the baby on their back. The goal is to survive as long as possible, avoiding obstacles along the path. As the speed increases over time, it becomes more difficult to avoid these obstacles, and if the player crashes into any, they lose the game.

Another key objective is to collect as many coins as possible. Gold coins are scattered throughout the path, and gathering them provides the baby with the wealth necessary to live a life among humans once he reaches his leadership age.

The game addresses the challenge of survival and resource collection under increasing pressure. As the player's speed escalates, it becomes progressively harder to avoid obstacles, simulating the difficulty of the journey while adding an element of urgency and decision-making.

The play values of this game are centered around quick reflexes, strategy, and resource management. The player must balance survival with the goal of collecting coins, offering a dynamic gameplay experience where speed, focus, and timing are essential to succeed.

Analysis

Resources:

The resources in the game consist of the baby on the wolf mother's back and the gold coins scattered along the path. The baby represents the ultimate goal of the game, with its survival and future development depending on how well the player performs. The gold coins are a secondary resource that adds a layer of strategic depth, allowing the player to collect wealth to help the baby later in life.

Boundaries:

The boundaries of the game are defined by the road the player must navigate and the obstacles that appear along the way. These obstacles include various environmental hazards, and colliding with any of them results in failure. Additionally, the game has a time-based boundary where the speed increases as time progresses, making it harder to survive and collect coins.

Interaction Patterns:

The main interaction pattern is that the player controls the wolf mother's movement, guiding her to avoid obstacles and collect coins. The game's pace and difficulty increase over time as the speed ramps up, forcing the player to make faster decisions and react quicker to changes in the environment. The interaction is direct, with the player's actions affecting the immediate outcome (whether the wolf survives or crashes).

Outcomes:

The primary outcome of the game is either survival or failure. If the player successfully navigates the road without crashing into obstacles and collects enough coins, they continue progressing. However, failure occurs when the player crashes into an obstacle, and the game ends. The wealth (coins) collected along the way contributes to the future of the baby, adding an element of strategy regarding resource accumulation.

Structural Conflict:

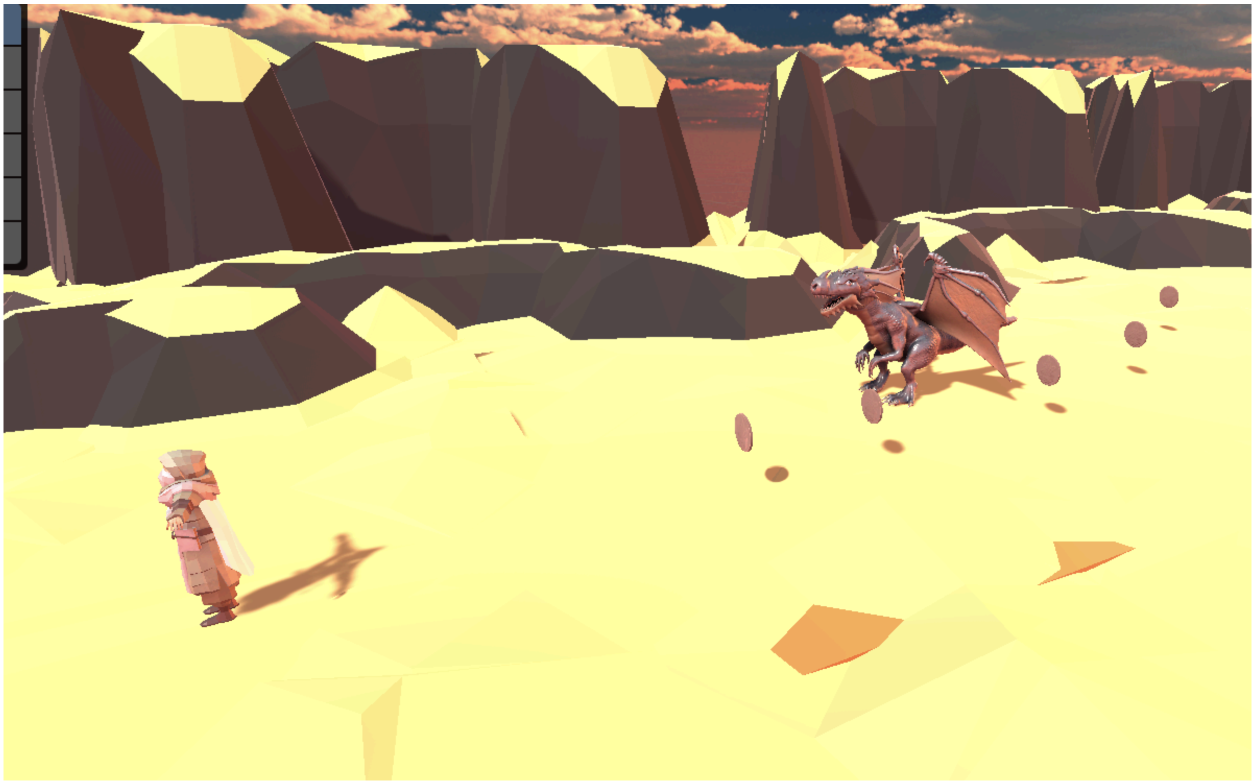
The structural conflict in the game arises from the need to balance two competing objectives: survival and collecting coins. As the game progresses, the speed increases, making it more difficult to avoid obstacles. The player must constantly make decisions on whether to focus on avoiding immediate obstacles or prioritize collecting coins for long-term success, creating a tension between short-term survival and long-term resource collection.

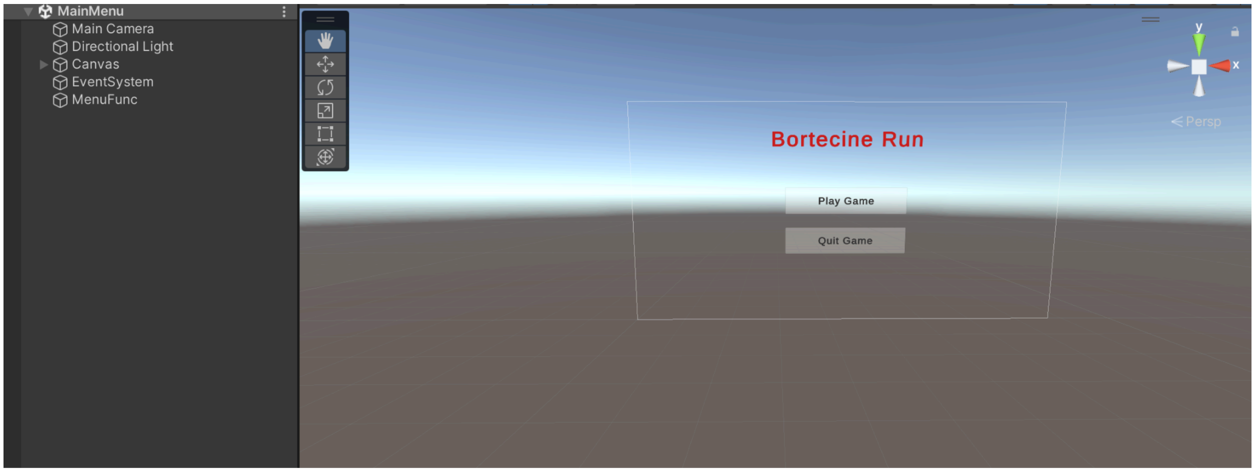
Game Theory Analysis:

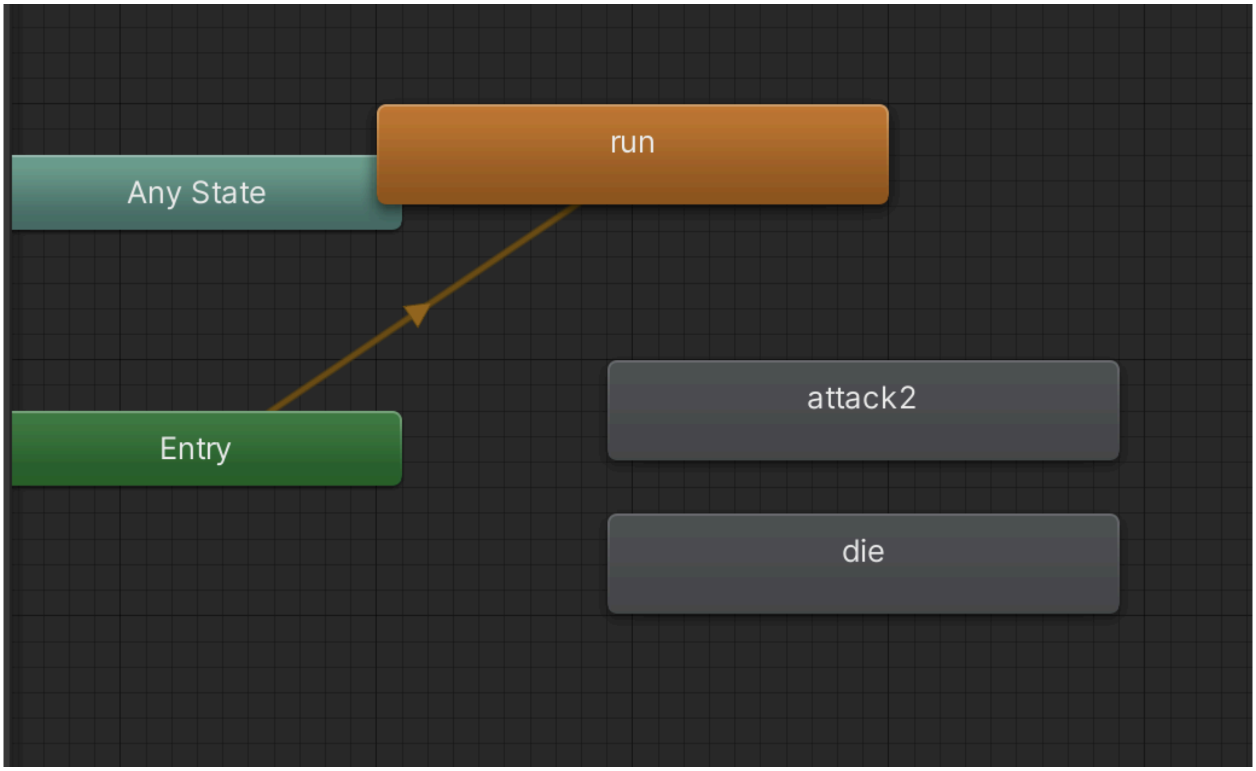
In terms of game theory, this game can be seen as a dynamic decision-making problem where the player must constantly evaluate risk versus reward. The increasing speed can be analyzed as a constraint that forces the player to adapt their strategy over time. The player's choices of when to focus on survival versus collecting coins can be modeled as a trade-off between immediate and delayed rewards. The game encourages a strategic approach to balance these competing goals under changing conditions, which mirrors elements of dynamic games where players must adjust their strategy based on evolving circumstances.

Design

Below, you will find the visual designs and mockups of the game, showcasing the main elements of gameplay, including the environment, obstacles, coin placements, and the wolf mother character. These designs illustrate the overall user interface (UI) and gameplay mechanics, providing a clear understanding of how the game is structured.




MainMenu
Main Camera
Directional Light
Canvas
EventSystem
MenuFunc
Bortecine Run
Play Game
Quit Game
y
x
Persp


run
Any State
attack2
Entry
die

Initial Design Idea

In the initial design phase, the game was built around the concept of survival under increasing difficulty, where the player controls a wolf mother carrying a baby on a straight road filled with obstacles. The core challenge selected for this game is managing the balance between survival (avoiding obstacles) and resource collection (gold coins). The reason for selecting this challenge is that it creates a dynamic and engaging experience, where the player must constantly adapt to the changing game conditions as the speed increases, adding complexity over time. This challenge is ideal because it tests both the player's reflexes and their strategic thinking.

The formal elements of the game are based on the concepts of action, timing, and resource management. The gameplay revolves around avoiding obstacles (collision avoidance) while collecting coins, and the game's increasing speed and randomly generated sections create a sense of tension and urgency.

The game logic follows a simple structure: the player must guide the wolf mother along a path, dodging obstacles while collecting coins. The random generation of road sections and obstacles ensures a fresh challenge each time, adding replayability. The game's premise is that the wolf mother must protect and raise the baby while navigating a dangerous, ever-changing environment. The story, rooted in Turkish mythology, emphasizes survival, resource accumulation, and the development of the baby into a leader.

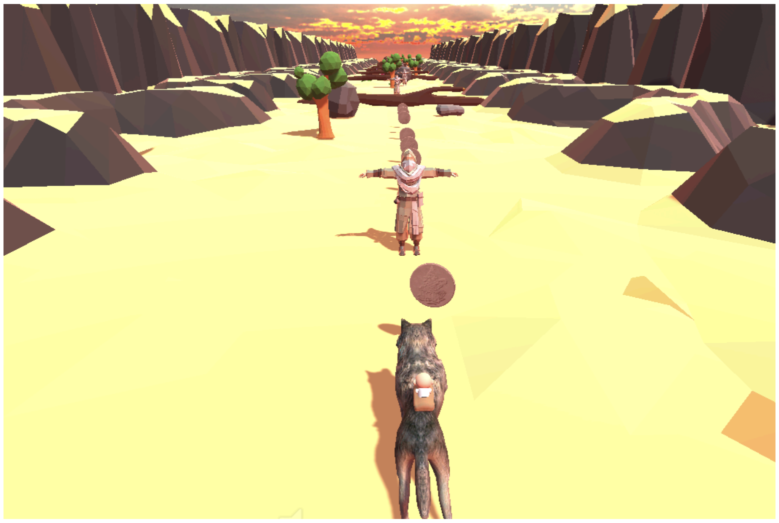
Final Design Idea

In the final design, the game mechanics, dynamics, and rules were refined to enhance player experience and ensure the game's intended challenge. The mechanics include random generation of road sections, with obstacles appearing unpredictably, creating an element of surprise and forcing the player to react quickly. As the game progresses, the speed increases, making it harder for the player to avoid obstacles and collect coins. The game follows a continuous cycle of random sections being generated, and once these sections are traversed, they are removed, ensuring that the gameplay environment remains dynamic.

The dynamics of the game revolve around the relationship between survival and coin collection. Players must make strategic decisions on when to prioritize survival (avoiding obstacles) and when to risk it for collecting coins. The rules are straightforward: the player controls the wolf mother's movements, and if the wolf crashes into an obstacle, the game ends. Coins collected during the journey contribute to the baby's future.

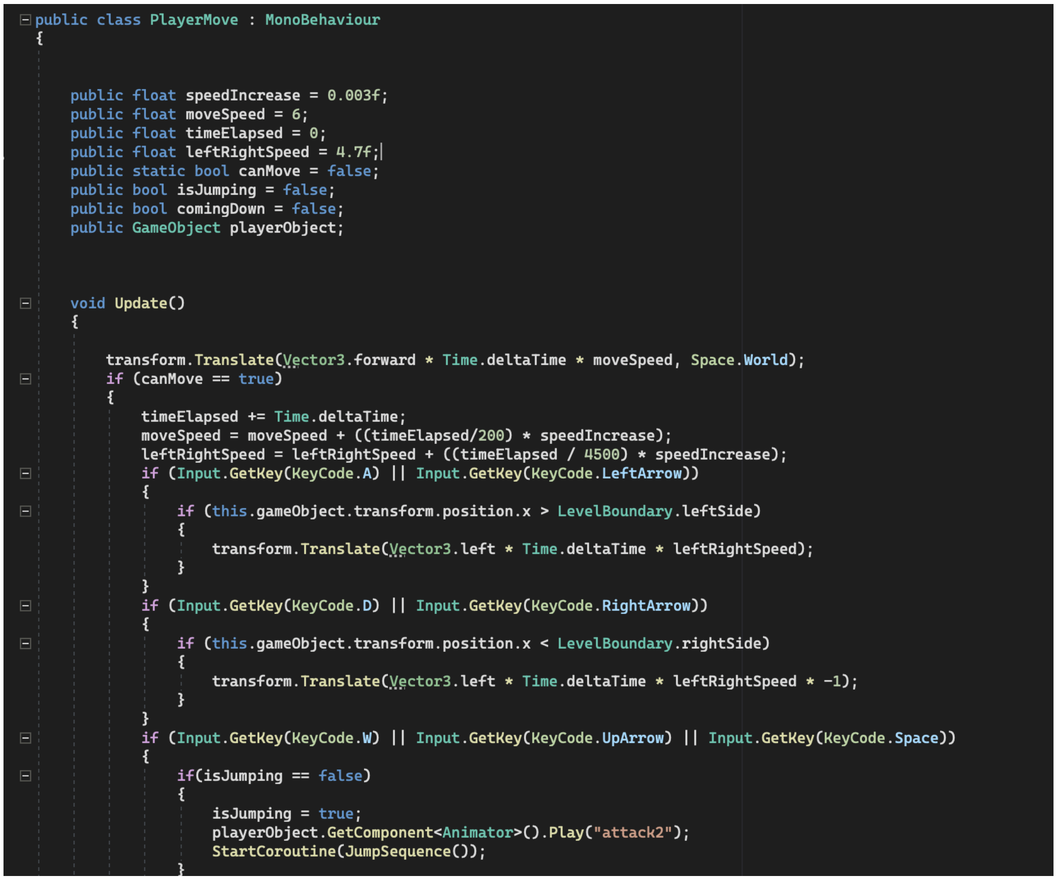
The game works by constantly presenting new challenges through random level generation and increasing difficulty. The player's actions directly impact the outcome, creating a feedback loop where every decision made influences the chance of survival and the success of collecting coins. As time progresses, the game introduces more obstacles and higher speed, ensuring a continuous escalation of difficulty, and thus, a more engaging and challenging experience.

Implementation



Start of the Game:

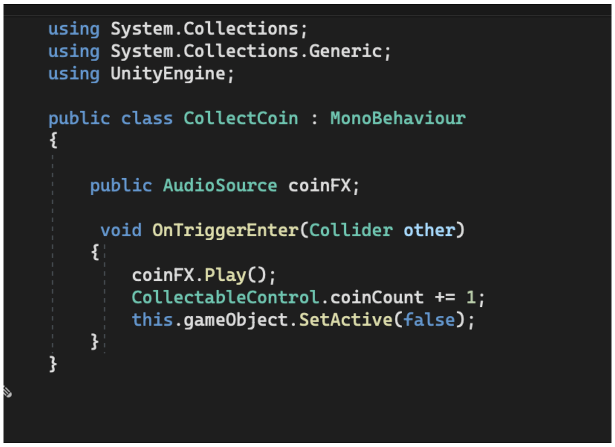
The player starts with the wolf running, carrying the baby. The primary objective is to navigate a path filled with obstacles and collect coins.



```
public class PlayerMove : MonoBehaviour
{


    public float speedIncrease = 0.003f;
    public float moveSpeed = 6;
    public float timeElapsed = 0;
    public float leftRightSpeed = 4.7f;
    public static bool canMove = false;
    public bool isJumping = false;
    public bool comingDown = false;
    public GameObject playerObject;



    void Update()
    {

        transform.Translate(Vector3.forward * Time.deltaTime * moveSpeed, Space.World);
        if (canMove == true)
        {
            timeElapsed += Time.deltaTime;
            moveSpeed = moveSpeed + ((timeElapsed/200) * speedIncrease);
            leftRightSpeed = leftRightSpeed + ((timeElapsed / 4500) * speedIncrease);
            if (Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.LeftArrow))
            {
                if (this.gameObject.transform.position.x > LevelBoundary.leftSide)
                {
                    transform.Translate(Vector3.left * Time.deltaTime * leftRightSpeed);
                }
            }
            if (Input.GetKey(KeyCode.D) || Input.GetKey(KeyCode.RightArrow))
            {
                if (this.gameObject.transform.position.x < LevelBoundary.rightSide)
                {
                    transform.Translate(Vector3.left * Time.deltaTime * leftRightSpeed * -1);
                }
            }
            if (Input.GetKey(KeyCode.W) || Input.GetKey(KeyCode.UpArrow) || Input.GetKey(KeyCode.Space))
            {
                if(isJumping == false)
                {
                    isJumping = true;
                    playerObject.GetComponent<Animator>().Play("attack2");
                    StartCoroutine(JumpSequence());
                }
```

This shows how the player moves the wolf on the road. The wolf accelerates over time, making it harder to avoid obstacles.

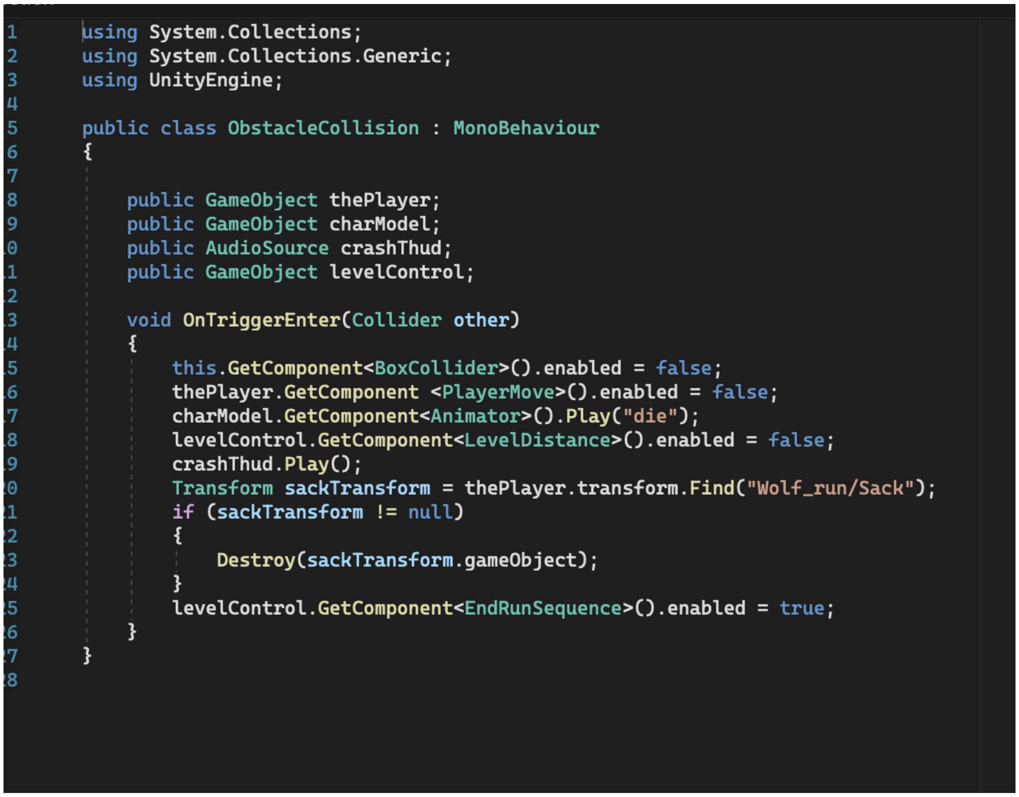
Coin Collection:



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

public class CollectCoin : MonoBehaviour
{

    public AudioSource coinFX;

     void OnTriggerEnter(Collider other)
    {
        coinFX.Play();
        CollectableControl.coinCount += 1;
        this.gameObject.SetActive(false);
    }
}
```

The player collects coins scattered along the path, contributing to the baby's future**.** The coin collection mechanism is controlled by triggering collisions between the player and the coin objects. When a coin is collected, the coin count increases, and a sound effect is played.

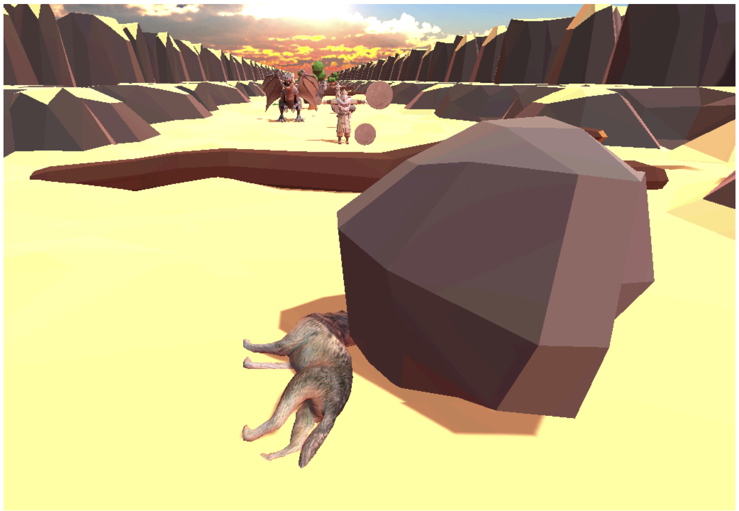
Collision with Obstacles:



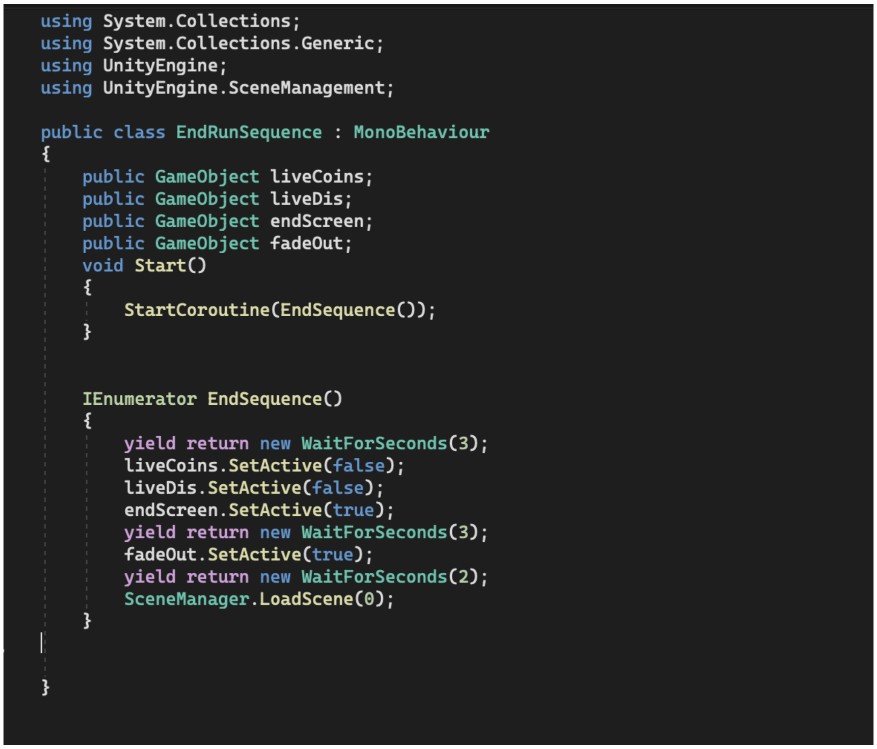
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ObstacleCollision : MonoBehaviour
{

    public GameObject thePlayer;
    public GameObject charModel;
    public AudioSource crashThud;
    public GameObject levelControl;

    void OnTriggerEnter(Collider other)
    {
        this.GetComponent<BoxCollider>().enabled = false;
        thePlayer.GetComponent <PlayerMove>().enabled = false;
        charModel.GetComponent<Animator>().Play("die");
        levelControl.GetComponent<LevelDistance>().enabled = false;
        crashThud.Play();
        Transform sackTransform = thePlayer.transform.Find("Wolf_run/Sack");
        if (sackTransform != null)
        {
            Destroy(sackTransform.gameObject);
        }
        levelControl.GetComponent<EndRunSequence>().enabled = true;
    }
}
```

The game becomes increasingly difficult as the speed of the wolf increases. If the wolf collides with an obstacle, the game ends, and the player is notified with a visual and audio cue. This code snippet shows the collision detection when the player hits an obstacle, triggering an animation, playing the crash sound, and stopping the player's movement.

End Game Sequence:



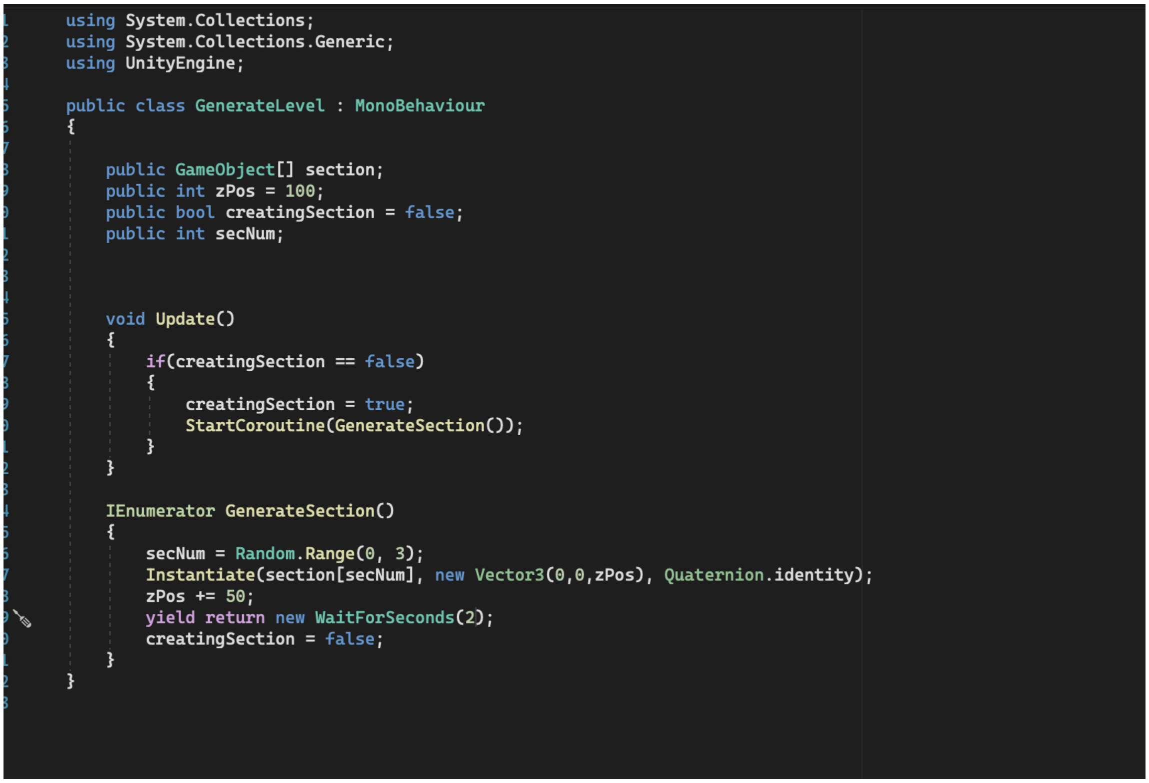
If the player collides with an obstacle, the game ends, and a sequence is triggered to display the end screen.



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

public class EndRunSequence : MonoBehaviour
{
    public GameObject liveCoins;
    public GameObject liveDis;
    public GameObject endScreen;
    public GameObject fadeOut;
    void Start()
    {
        StartCoroutine(EndSequence());
    }


    IEnumerator EndSequence()
    {
        yield return new WaitForSeconds(3);
        liveCoins.SetActive(false);
        liveDis.SetActive(false);
        endScreen.SetActive(true);
        yield return new WaitForSeconds(3);
        fadeOut.SetActive(true);
        yield return new WaitForSeconds(2);
        SceneManager.LoadScene(0);
    }


}
```

This section of the game handles the end sequence, including hiding coins, showing the game over screen, and transitioning to the next scene after a brief delay.

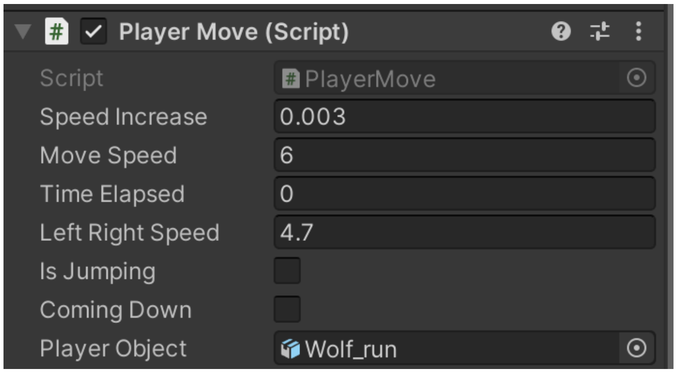
Section Generation:



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

public class GenerateLevel : MonoBehaviour
{

    public GameObject[] section;
    public int zPos = 100;
    public bool creatingSection = false;
    public int secNum;



    void Update()
    {
        if(creatingSection == false)
        {
            creatingSection = true;
            StartCoroutine(GenerateSection());
        }
    }

    IEnumerator GenerateSection()
    {
        secNum = Random.Range(0, 3);
        Instantiate(section[secNum], new Vector3(0,0,zPos), Quaternion.identity);
        zPos += 50;
        yield return new WaitForSeconds(2);
        creatingSection = false;
    }
}
```

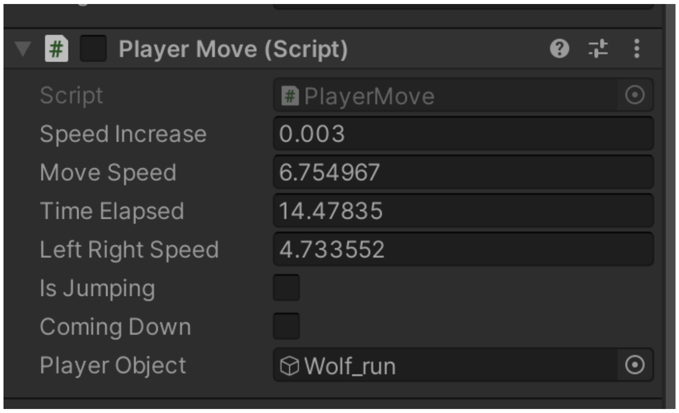
The game continuously generates new sections of the road as the wolf moves forward, ensuring that the player faces new obstacles. The game generates new sections randomly as the player progresses, making the game environment dynamic and unpredictable.

Movement and Speed:



☑ Player Move (Script)

Script	PlayerMove
Speed Increase	0.003
Move Speed	6
Time Elapsed	0
Left Right Speed	4.7
Is Jumping	☐
Coming Down	☐
Player Object	Wolf_run

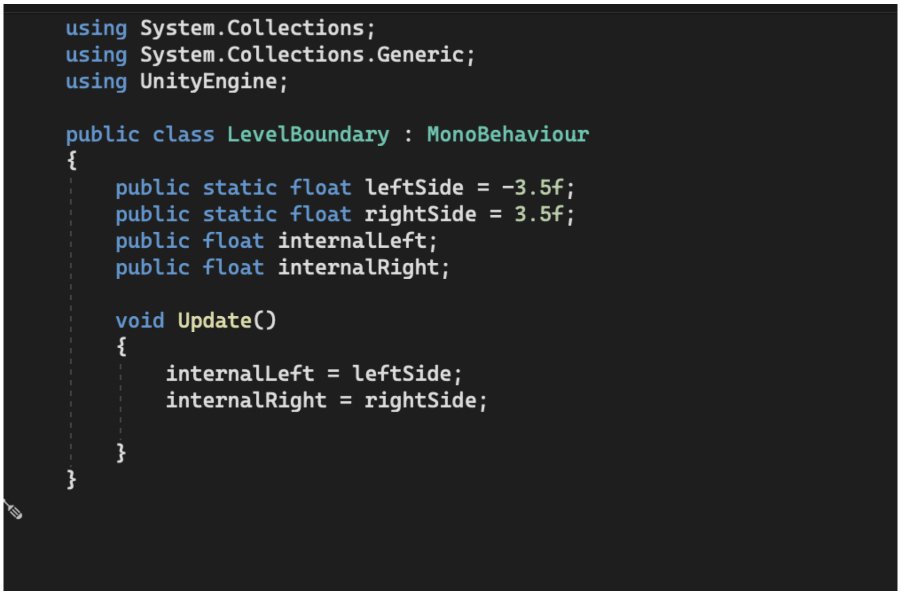


☐ Player Move (Script)

Script	PlayerMove
Speed Increase	0.003
Move Speed	6.754967
Time Elapsed	14.47835
Left Right Speed	4.733552
Is Jumping	☐
Coming Down	☐
Player Object	Wolf_run

The player can control the wolf's movement, including left and right motion. As time progresses, the speed increases, adding complexity. These screenshots show the progressive speed increase, which is controlled based on the elapsed time, making it harder for the player to react as the game continues.

Boundary for Movement:



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

public class LevelBoundary : MonoBehaviour
{
    public static float leftSide = -3.5f;
    public static float rightSide = 3.5f;
    public float internalLeft;
    public float internalRight;

    void Update()
    {
        internalLeft = leftSide;
        internalRight = rightSide;

    }
}
```

The game has boundaries that the player cannot cross. These boundaries help keep the wolf on the road and prevent the player from moving out of the playable area. This shows the level's boundaries and how the player's movement is restricted within them.

Evaluation

To evaluate the game prototype, a **focus group study** was conducted with two participants, both developers, including myself and my groupmate. The goal was to gather feedback on the gameplay experience, mechanics, and overall engagement from a developer's perspective.

Focus Group Details:

- **Participants:** The focus group consisted of 2 developers: myself and my groupmate, both with experience in game development and an understanding of casual games. We played the game to provide insights on the game's functionality, challenge balance, and user experience.

- **Study Duration:** Both participants played the game for approximately 20 minutes each, focusing on core mechanics like navigation, obstacle avoidance, and coin collection.
- **Methodology:** The study was conducted in a collaborative setting, where we played the game individually and then discussed the experience afterward to gather feedback.

Key Questions:

- How engaging did you find the game?
- Were the controls intuitive and easy to understand?
- How did you feel about the increasing speed and its impact on gameplay?
- Did you experience any frustration or confusion while playing?
- How would you rate the game's visual and audio elements?
- Would you be motivated to play this game again?

Findings:

- **Gameplay Mechanics:** Both developers found the core gameplay mechanics interesting, particularly the balance between obstacle avoidance and coin collection. However, the rapid increase in speed was noted as a challenge, with both participants suggesting a more gradual speed increase for better player control. This would ensure a more balanced difficulty curve.
- **Controls:** The controls were easy to grasp, though the initial increase in speed created moments of confusion, especially when trying to avoid obstacles at higher speeds. Both participants felt that with better gradual learning curves and visual cues, the controls could be even more intuitive.
- **Difficulty Balance:** Both developers agreed that the sudden increase in speed could be overwhelming and recommended adjusting the difficulty curve. Introducing adjustable difficulty settings or smoother transitions between stages could enhance player experience and accommodate different skill levels.
- **Visual and Audio Feedback:** The game's visual style, including road design and obstacle placement, was found to be functional and visually appealing. The audio elements, such as sound effects for collecting coins and crashing into obstacles, were recognized as adding a fun and immersive touch to the gameplay.
- **Replayability:** The random generation of levels was appreciated, providing replay value. However, both participants suggested adding more diverse obstacles or interactive elements (such as power-ups) to increase engagement in repeated playthroughs.

Conclusion: The focus group study provided useful insights, especially regarding the balance of game mechanics and difficulty progression. Based on the feedback, the next steps will focus on refining the difficulty curve, introducing more variety in obstacles, and providing clearer guidance for the player during the initial stages of gameplay.

Results of Your Play Testing Session

During the playtesting session, several observations were made regarding the gameplay experience, mechanics, and overall performance. Both participants tested the game for approximately 20 minutes each, focusing on the key aspects such as movement, obstacle collision, and speed progression.

Key Observations:

1. **Collision Boundaries:**
    - It was observed that some of the collision boundaries for the obstacles could benefit from optimization. In certain situations, the detection of collisions was either too sensitive or not responsive enough, leading to unexpected deaths or missed collisions. This could be adjusted to provide a smoother, more predictable experience for the player.
2. **Speed Increase:**
    - The speed increase feature worked as intended but could be further optimized. The rate at which the speed increases could be made smoother to avoid sudden difficulty spikes, making it easier for players to adjust. This would create a more natural progression in terms of challenge without overwhelming the player too quickly.
3. **Jump Animation:**
    - While the jump mechanics functioned properly, it was noted that the jump animation could be enhanced for better visual feedback. The animation was somewhat basic, and improving its fluidity could increase the player's immersion and provide more intuitive visual cues when the wolf jumps.

Other Notable Observations:

- The random level generation was effective in maintaining gameplay variability, though more obstacles or environmental features could add variety to the experience.
- Audio cues, such as the coin collection and crash sounds, were appropriately timed and enhanced the overall gameplay experience

Recommendations & Conclusions

1. **Optimize Collision Boundaries:**
    - Improve the collision detection system to make it more responsive and accurate. The current collision boundaries sometimes cause unintended deaths or miss collisions. Fine-tuning the collider parameters and ensuring they align more consistently with the obstacles could create a smoother experience for players.
2. **Smooth Speed Increase:**

   - The speed increase feature could be optimized to avoid sudden difficulty spikes. Implementing a more gradual speed increase or introducing adjustable difficulty settings would allow players to adjust the challenge level to their preference, improving the overall player experience and making the game accessible to a wider audience.

3. **Enhance Jump Animation:**
   - The jump animation could be improved for better fluidity and visual feedback. A smoother, more dynamic animation would make the jumping action feel more realistic and satisfying, and provide clearer cues to the player on the wolf's movements.
4. **Increase Environmental Variety:**
   - Adding more varied obstacles and interactive elements such as power-ups or additional hazards would make the gameplay more engaging and challenging. This would also help to maintain replayability, as players would encounter different situations in each session.
5. **Polish User Interface (UI):**
   - While the game is functional, improving the UI could enhance the overall experience. Clearer visual indicators for the coin count, health, and objectives, along with smoother transitions between menus and game screens, would make the game more polished.

Conclusions – What Went Well/Wrong and Why?

What Went Well:

- **Core Gameplay Mechanics:** The fundamental mechanics of the game, such as avoiding obstacles and collecting coins, were well-received. The feedback from the playtesting session indicated that the basic concept of balancing survival and resource collection was engaging.
- **Random Level Generation:** The random generation of road sections worked effectively, keeping the game fresh and providing variability in each playthrough.
- **Audio & Visual Design:** The audio cues for coin collection and crashes, along with the basic visual design of the road and obstacles, added to the immersion and enjoyment of the game.

What Went Wrong:

- **Collision Boundaries:** Some of the collision boundaries caused unexpected deaths or allowed the player to pass through obstacles without any consequences. This resulted in moments of frustration for the players.
- **Speed Increase:** The sudden speed increase, especially as the game progressed, created difficulty spikes that were not well-received by all players. This made the game feel too challenging too quickly, causing some players to become frustrated.
- **Jump Animation:** The jump animation was functional but lacked fluidity, which could hinder the player's experience by making the jump mechanics feel stiff or unnatural.

Why These Issues Occurred:

- The collision boundary issues likely stemmed from misalignment between the visual representation of obstacles and their collision detection areas. This can be fixed by revising the collision box sizes and positions.
- The speed increase, while effective in raising difficulty, was too abrupt because it was designed to ramp up quickly without gradual transitions. This created an imbalance in difficulty.
- The jump animation was a simple placeholder, and its lack of smoothness resulted from not having a fully refined animation system.

References

https://www.youtube.com/watch?v=N8QMuFPK4v0&t=55s
https://www.youtube.com/watch?v=U00zyEi86A8
https://forum.gdevelop.io/t/how-can-i-make-an-endless-runner-more-optimized/36491
https://docs.unity3d.com/Manual/AnimationSection.html