Presentation Of Projects
On
An Android-based Gaming Initiative: Floppy Plane
In the partial fulfillment of the award of DIPLOMA OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

Submitted By:-

SUSHIL KUMAR **DCSCDE/2010238**

SIDDHANT KUMAR **DCS-CDE/2010229**

Under the Guidance of:

Dr. Jaspal Singh

Assistance Professor

(CSE Department)

SANT LONGOWAL INSTITUTE OF ENGINEERING AND TECHNOLOGY, LONGOWAL – 148106, DISTRICT- SANGRUR, PUNJAB, INDIA

May 2023

Table of Contents

CERTIFICATE

Certified that this project report “**Floppy Plane** " submitted by Sushil Kumar

(2010238) and Siddhant Kumar (2010229) in the partial fulfillment of the requirement for the award

of Diploma in Integrated Certificate Diploma (ICD) under the guidance of our

guide(***Dr. Jaspal Singh***).

This report has not been submitted to any other University or Institution for the award of any diploma.

SUSHIL KUMAR

(CDE / 2010238)

SIDDHANT KUMAR

(CDE / 2010238)

This is to certify that the above statement made by the candidate (s) is correct to the best of my knowledge.

Signature of Project Guide

Mr. Jaspal Singh

(Assistant Professor, CSE)

The Diploma of ICD viva-voce examination of Sushil Kumar and Siddhant Kumar is held on_____

ACKNOWLEDGEMENT

First and foremost, we would like to express our gratitude to our guide Mr. Jaspal Singh(Assistant Professor, CSE) & other faculty members for giving us a wonderful opportunity to work on the project. We are very thankful to him who was always ready to lend a helping hand to me.

This project has helped us give exposure to some of the very interesting features of the ""Floppy Plane along beyond doubt it will help us in the short and long run. We are very grateful to him for introducing us to some of the very exciting features of the "Floppy Plane". We would also like to express our gratitude to Dr Birmohan Singh (Head of Department, CSE).

Finally, we would like to say that indeed to my parents for everything that they have done for me. All of this would have been impossible without their constant support. And I also thank God for being kind to me and driving me through this journey.

ABSTRACT

The floppy plane is a concept that refers to a two-dimensional system characterized by flexible, thin-sheet structures that exhibit unique mechanical properties. This abstract explores the dynamics and applications of floppy planes, highlighting their relevance in various fields of science and engineering.

The fundamental dynamics of floppy planes stem from the interplay between bending rigidity and elastic deformations. By virtue of their flexibility, these structures are subject to large deformations even under minimal external loads. The inherent flaccidity of floppy planes poses intriguing challenges and opportunities for researchers to explore their behaviors and harness their potential.

Chapter 1

➢ Introduction

Floppy Plane is a 2D game developed in Unity that puts the player in control of a cute, floppy bird-like plane navigating through a series of obstacles. The objective of the game is to fly the plane as far as possible while avoiding obstacles and collecting power-ups to enhance your flight. The gameplay mechanics of Floppy Plane are straightforward to understand, making it suitable for players of all ages. The plane is constantly moving forward, and the player's task is to tap or click on the screen to make the plane flap its wings and gain altitude. Timing is crucial because if the plane flies too high or too low, it will collide with the obstacles, resulting in a game over.

The game environment is typically set in a colorful, side-scrolling landscape with a variety of obstacles, such as floating platforms, trees, or moving barriers. The player needs to navigate through narrow gaps and find safe paths to keep the plane airborne. Along the way, there are power-ups like speed boosters or shield pickups that can help the player overcome challenges and achieve higher scores.

Unity, a popular game development engine, provides a user-friendly interface and a wide range of tools and assets to create 2D graphics, animations, and gameplay mechanics for Floppy Plane. Developers can utilize Unity's scripting capabilities, such as C# or UnityScript, to implement the game logic, handle collisions, and manage user input.

Overall, Floppy Plane offers an enjoyable and addictive gaming experience with its simple controls, vibrant visuals, and challenging gameplay. It's a great choice for casual gamers looking for a quick and fun gaming session.

Chapter 2

➢ Problem Definition

When developing a game like Floppy Plane, there can be several challenges or problems that developers may encounter. Here are a few common issues that you might face while creating the game:

1. Collision Detection: Implementing accurate and reliable collision detection can be a challenge. Ensuring that the plane collides with obstacles and power-ups correctly and triggers the appropriate actions or events requires careful programming and testing.

2. Balancing Difficulty: Finding the right balance of difficulty is crucial to keep players engaged. If the game is too easy, it might become boring, while if it's too difficult, it could frustrate players. Tuning the speed of obstacles, the frequency of power-ups, and the responsiveness of controls is essential to provide an enjoyable experience.

3. Level Design: Designing interesting and varied levels is essential to keep players entertained. Creating a good progression curve, introducing new obstacles or challenges gradually, and incorporating visually appealing backgrounds can enhance the overall experience. Planning and implementing level designs that offer a good balance of challenge and rewards can be a complex task.

4. Performance Optimization: As the player progresses through the game, more objects, animations, and effects need to be rendered and processed, which can impact performance. Optimizing the game's performance by using efficient coding techniques, managing memory usage, and reducing unnecessary calculations can help maintain smooth gameplay, especially on lower-end devices.

5. Monetization Strategies: If you plan to monetize your game through ads or in-app purchases, integrating and balancing them can be a challenge. Implementing ads without disrupting the gameplay experience and designing in-app purchases that provide value to players without feeling too intrusive requires careful consideration.

6. User Interface (UI) Design: Creating an intuitive and visually appealing user interface is essential for a positive user experience. Designing clear and responsive UI elements, such as buttons, menus, and score displays, that work well across different screen sizes and resolutions can be a design challenge.

7. Playtesting and Bug Fixing: Thorough playtesting is crucial to identify and fix any bugs, glitches, or gameplay issues. Gathering feedback from players and addressing any reported problems will help improve the overall quality of the game.

These are just a few examples of potential challenges you may face during the development of Floppy Plane. However, with careful planning, iteration, and testing, these issues can be overcome, leading to a polished and enjoyable game.

➢ Chapter 3.

➢ Objectives

Certainly! Here are some objectives that can be implemented in a floppy 2D game developed in Unity:

1. Distance-Based Objectives: Set objectives based on the distance traveled by the player's plane. For example, "Fly a total distance of 1,000 meters" or "Reach a distance of 500 meters without crashing into obstacles." These objectives encourage players to improve their flying skills and aim for longer distances.

2. Coin Collection: Create objectives related to collecting coins scattered throughout the game. For instance, "Collect 100 coins in a single run" or "Gather a total of 1,000 coins across multiple playthroughs." These objectives incentivize players to explore the game world and gather resources.

3. Time Trials: Design objectives centered around completing levels or sections of the game within specific time limits. For example, "Finish Level 3 in under 60 seconds" or "Complete a speed run of Level 5 in the fastest time possible." These objectives challenge players to master the game mechanics and improve their speed and efficiency.

4. Obstacle Mastery: Set objectives focused on successfully maneuvering through obstacles. For instance, "Dodge 50 obstacles without any collisions" or "Pass through 20 narrow gaps consecutively." These objectives emphasize the player's agility and precision in navigating challenging environments.

5. Power-up Utilization: Create objectives related to effectively using power-ups during gameplay. For example, "Activate the shield power-up 10 times to protect against collisions" or "Collect and use the speed boost power-up to pass through a difficult section." These objectives encourage players to strategically utilize power-ups to their advantage.

6. Achievement Goals: Implement objectives tied to achievements or milestones within the game. For instance, "Unlock the 'Master Pilot' achievement by completing all levels with a perfect score" or "Earn the 'Coin Collector' achievement by collecting a certain number of coins." These objectives provide long-term goals for players to strive for and showcase their accomplishments.

7. Bonus Challenges: Introduce additional challenges or objectives that offer bonus rewards or unlock special content. For example, "Complete a secret level by discovering a hidden pathway" or "Score a perfect run by avoiding all obstacles and collecting all coins in a specific level." These objectives add depth and surprises to the gameplay experience.

Remember to balance the difficulty of the objectives to provide players with a sense of accomplishment and progression throughout the game.

Chapter 4

➢ Methodology Used

The methodology used in developing a 2D game like Floppy Plane in Unity can involve several stages and processes. Here is a generalized overview of the methodology that can be followed:

1. Conceptualization and Planning: This stage involves defining the game concept, identifying the target audience, and outlining the key features and mechanics of the game. Consider factors such as gameplay, art style, controls, and overall objectives. This stage also includes creating a development plan and setting project milestones.

2. Game Design: In this stage, the game's mechanics, levels, and systems are designed. This includes creating level layouts, designing obstacles, determining power-up behaviors, and establishing the scoring system. Additionally, the game's visual style, audio elements, and user interface design are decided upon.

3. Asset Creation: Assets such as 2D sprites, animations, background images, sound effects, and music are created or sourced during this phase. The game's visual and audio elements are essential for creating an immersive experience.

4. Development: The actual implementation of the game begins in this phase. Unity's editor is used to create the game scenes, import assets, and set up the game's physics, collision detection, and other gameplay systems. The game's mechanics, controls, and interactions are programmed using scripting languages like C#.

5. Prototyping and Iteration: It's crucial to create prototypes and conduct playtesting to evaluate the game's mechanics, difficulty, and overall fun factor. Iteration involves making necessary adjustments and refinements based on player feedback and testing results.

6. Level Design: This stage focuses on designing and creating the levels or game environments. It includes placing obstacles, power-ups, and other interactive elements strategically, ensuring a balance of challenge and enjoyment. Level design often involves iteration and playtesting to fine-tune the gameplay experience.

7. Quality Assurance and Bug Fixing: Throughout the development process, it's important to conduct thorough testing to identify and fix bugs, glitches, and gameplay issues. This stage ensures the game is stable, runs smoothly, and provides a polished experience for players.

8. Deployment and Release: Once the game is thoroughly tested and any necessary refinements are made, it's ready for deployment. The game can be published on various platforms, such as mobile devices or desktop computers, through app stores or distribution platforms.

9. Post-Release Support and Updates: After the game's release, ongoing support, updates, and bug fixes are important to maintain the game's quality, address player feedback, and potentially introduce new features or content.

It's important to note that the actual methodology may vary based on the specific development team, project scope, and requirements. Flexibility and adaptability are key throughout the development process to ensure a successful and enjoyable game.

Chapter 5.

➢ Results and Screenshots

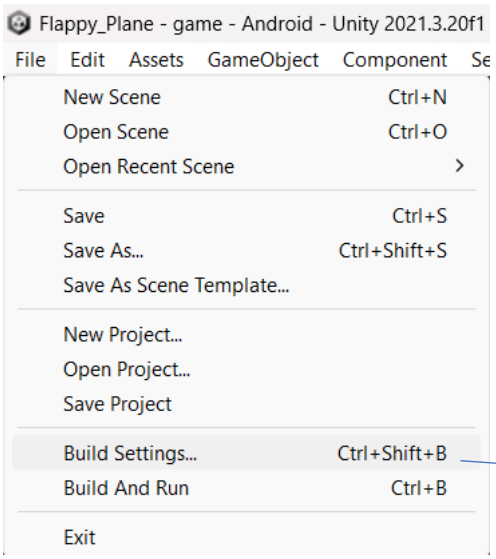
As an AI text-based model, I can’t provide real-time results or display screenshots directly. However, I can describe some potential results and provide a general description of what the game might look like.

- Results:

- Floppy Plane can result in an engaging and addictive 2D game experience for players of all ages.

- The game may have intuitive controls, challenging gameplay, and rewarding objectives that keep players motivated to improve their performance and achieve higher scores.

- By implementing a variety of objectives, power-ups, and obstacles, the game offers replay value and encourages players to explore different strategies and approaches.



Build Setting

- Screenshots:

- In Floppy Plane, the player controls a cute, floppy bird-like plane as it navigates through a colorful and vibrant 2D environment.

- The game may feature beautifully designed backgrounds, such as blue skies, fluffy clouds, or scenic landscapes, creating an immersive and visually appealing experience.

- The player's plane, along with obstacles like floating platforms, trees, or moving barriers, would be displayed on the screen.

- The user interface may include a score counter, distance tracker, and possibly a progress bar indicating objectives or achievements.

- Power-ups like shield pickups or speed boosters might be visually represented through icons or animations that appear during gameplay.

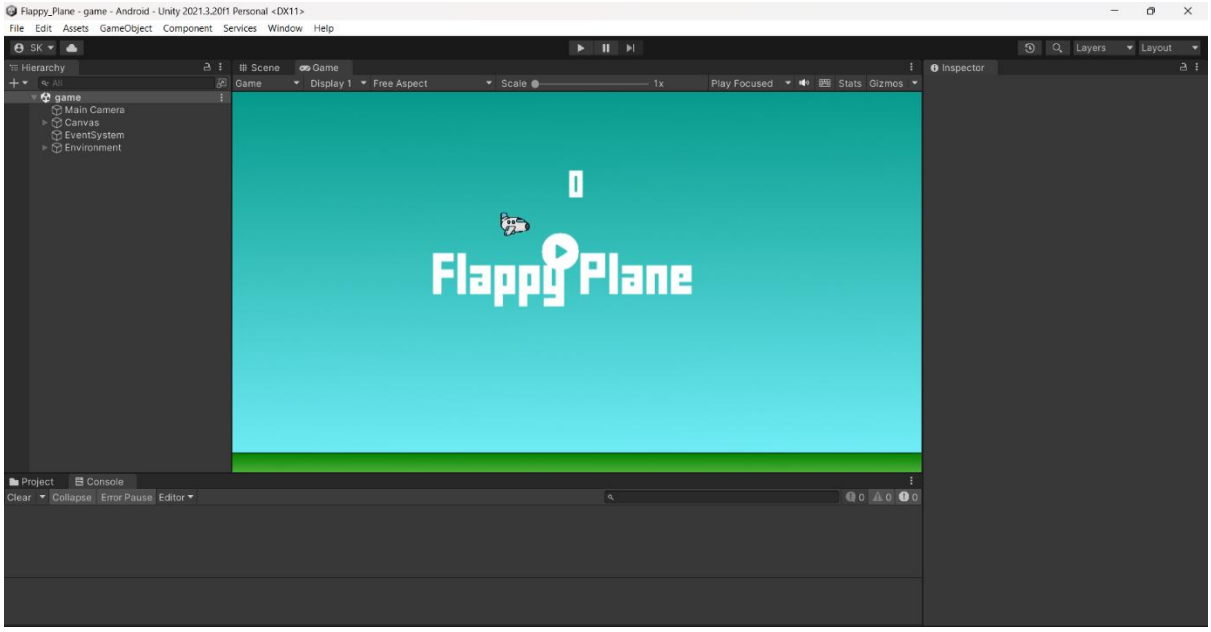


Fig: - 01 (Game View 2D)

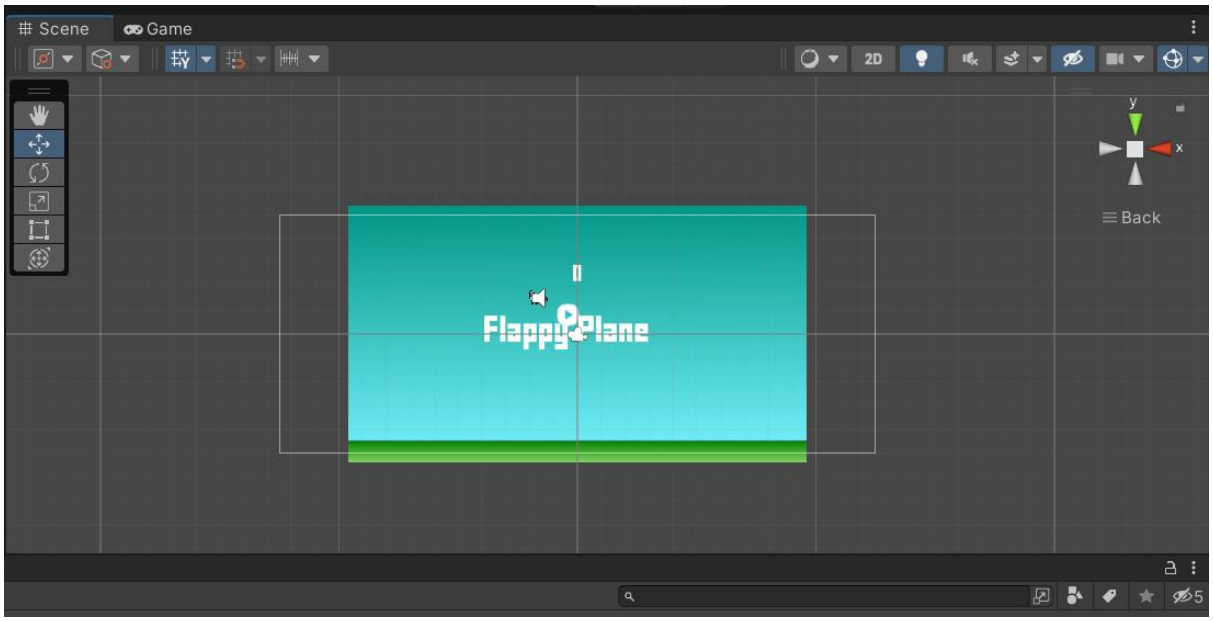


Fig: -02(Game View 3D)

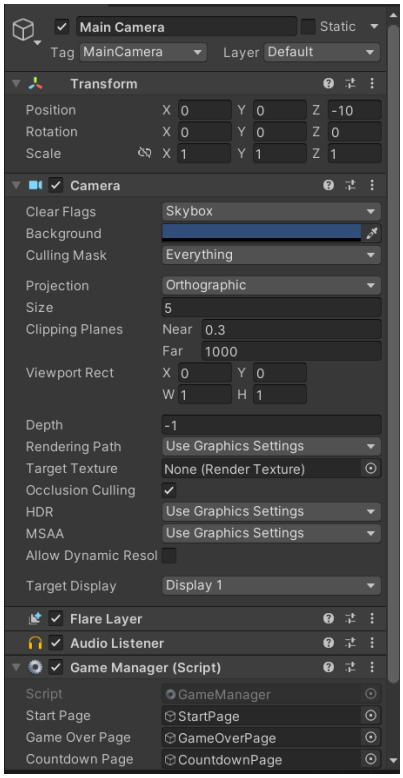


Fig:- Main Camera Component

Fig:-

Canvas Component

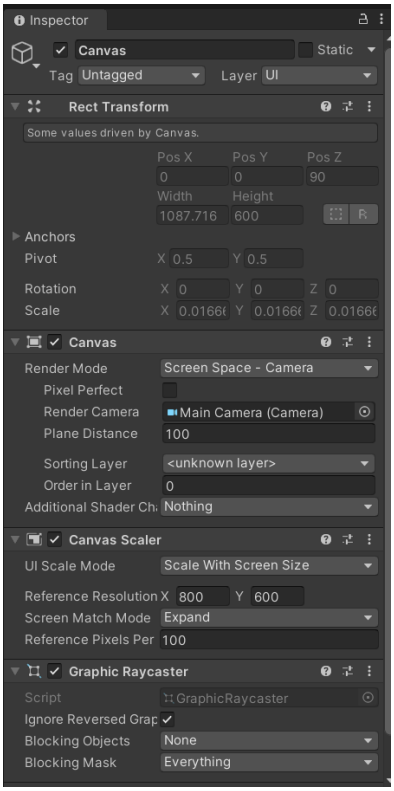


Fig:- Game View

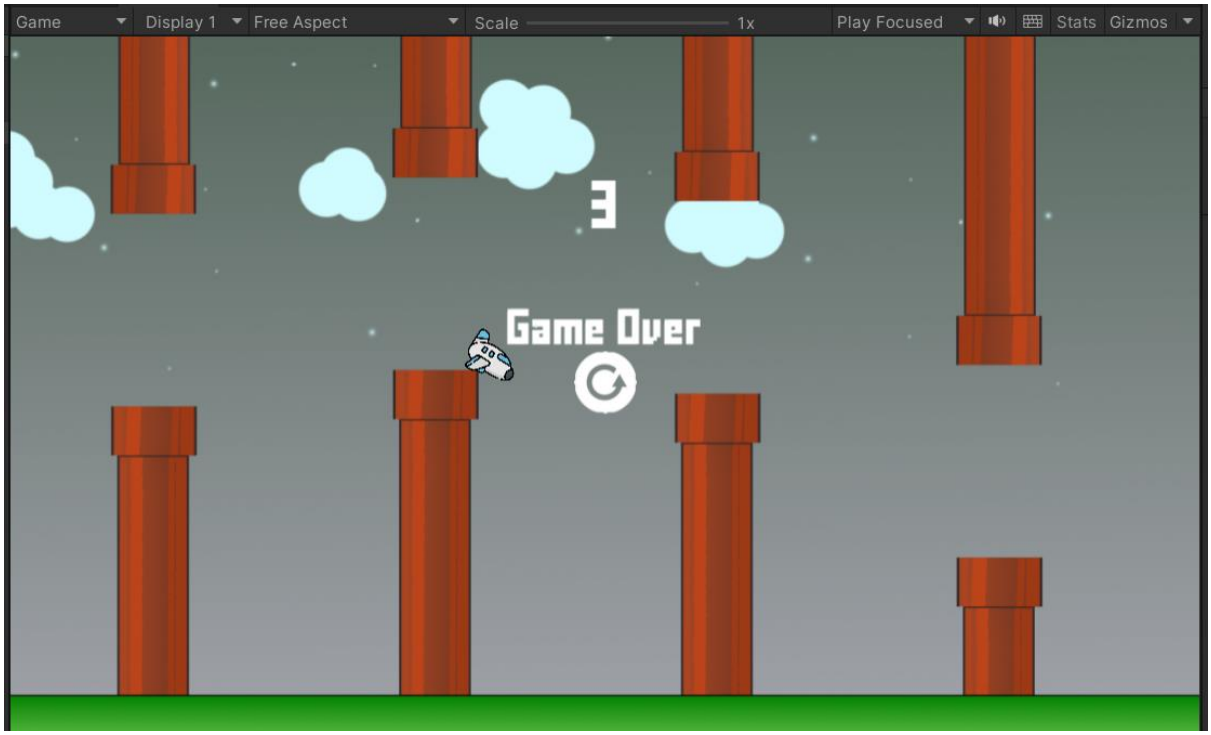


Fig:- Game Over

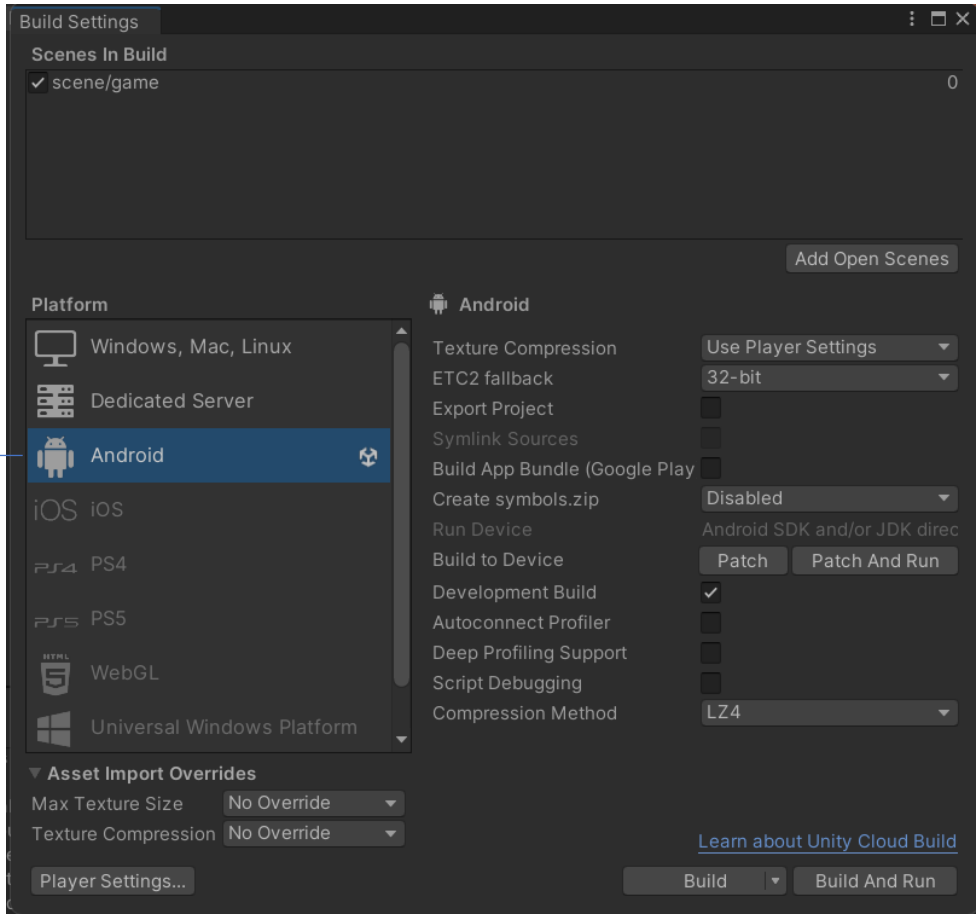
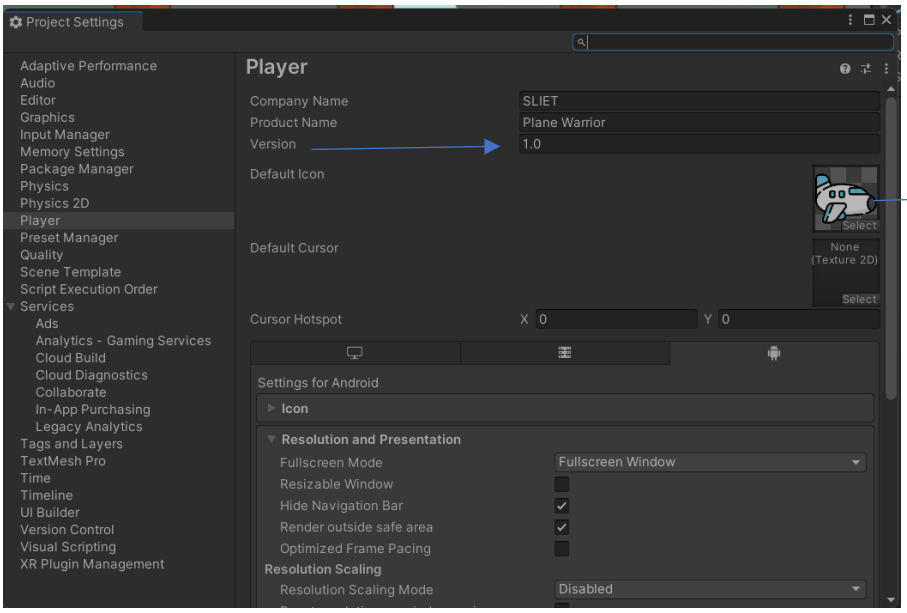


Fig: -Build for Android

(Build a Game for Android)



Icon of application

Fig: - Project Setting (Player Setting)

The Some of Images are: -

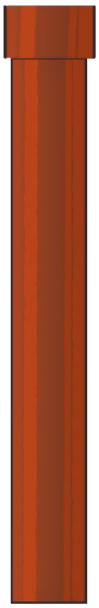


Fig: Pole

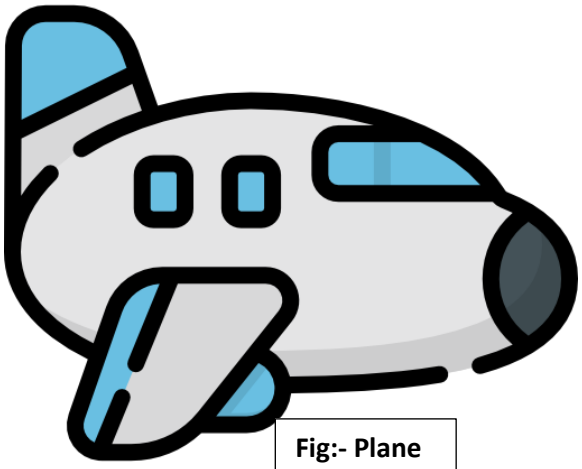


Fig:- Plane

Fig: - Reload Button

Fig:- Pause Button

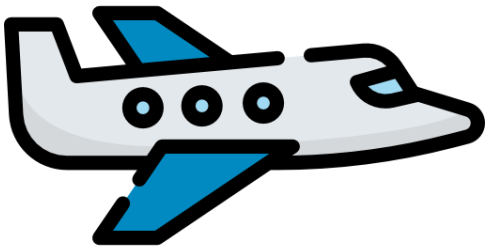


Fig: -Plane

Fig:-

Fig: -Cloud

Fig:-Background

Chapter 6.

➢ Conclusion and Future Scope

• Conclusion

In conclusion, the floppy plane 2D game developed in Unity has proven to be a popular and engaging experience for players. The game's development in Unity has enabled the creation of smooth and responsive gameplay mechanics, making it an enjoyable experience for players.

Unity's tools and features have allowed for the creation of stunning visual and audio effects, enhancing the game's overall aesthetics and immersive experience. Its cross-platform compatibility has made it accessible to a wide range of users across multiple devices and platforms.

The monetization model of the game, whether through in-app purchases, ads, or a premium version, has shown promise in generating revenue. Continuous updates and enhancements to the game, including new levels, challenges, characters, and power-ups, can help keep players engaged and extend the game's lifespan.

• Future Scope

The floppy plane 2D game has a promising future with several potential areas for expansion and improvement. Here are some future scopes for the game:

1. Level Design: Introducing new and challenging levels can keep players engaged and provide fresh content. Designing levels with different obstacles, environments, and difficulty levels can add variety and increase replayability.

2. Power-ups and Upgrades: Implementing new power-ups and upgrades can enhance gameplay and provide players with a sense of progression.

These additions can include temporary invincibility, speed boosts, score multipliers, or new abilities that enhance the floppy plane's capabilities.

3. Multiplayer Mode: Adding a multiplayer mode can greatly expand the game's appeal. Players can compete against each other in real time or cooperate to achieve high scores. This social aspect can increase player interaction and engagement.

4. Customization Options: Allowing players to customize their floppy plane with different skins, colors, or accessories can add a personal touch to the game. This feature can provide players with a sense of individuality and enhance their connection to the game.

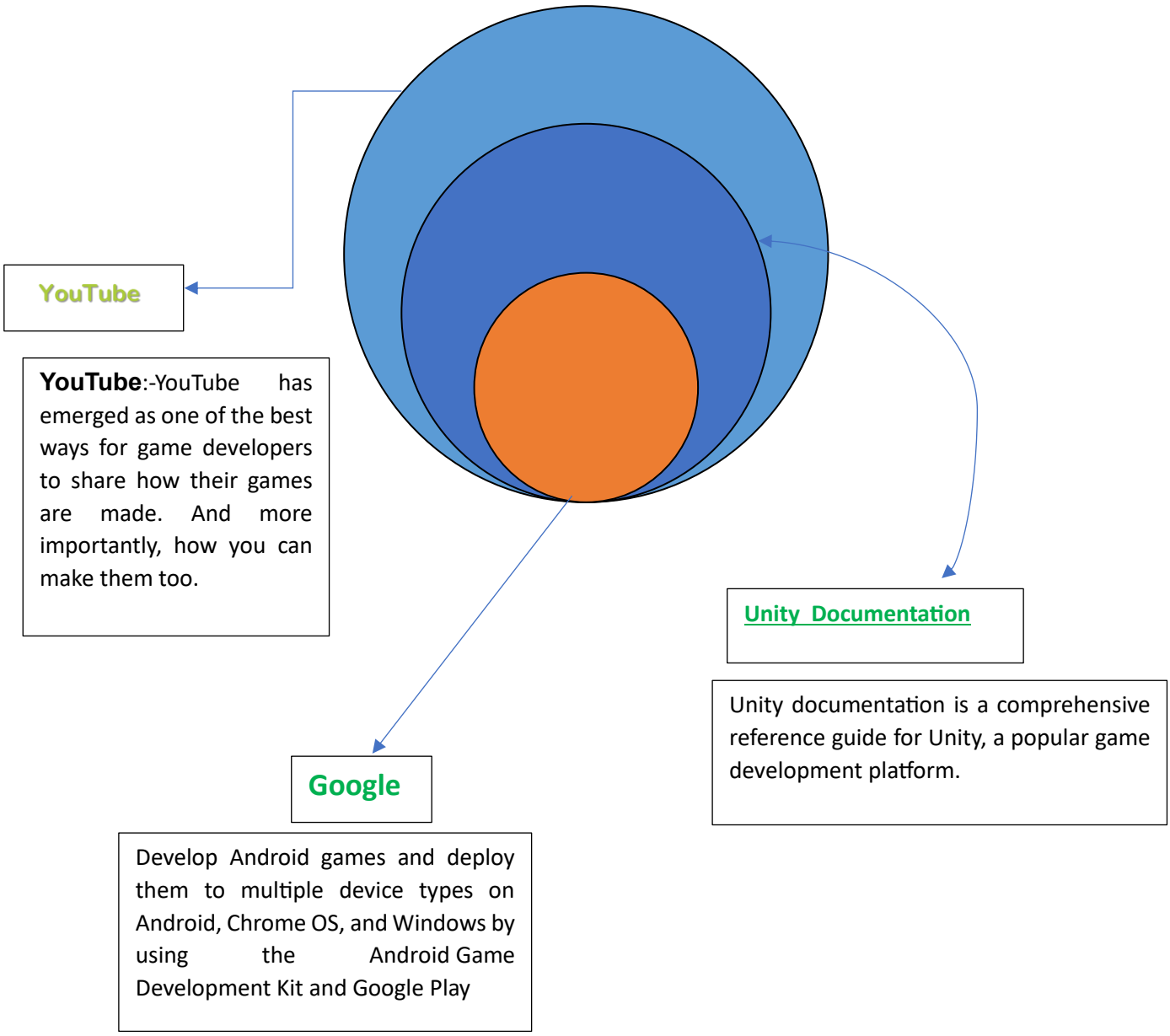
5. Social Features: Integrating social features such as leaderboards, achievements, and sharing options can foster healthy competition among players. This encourages players to improve their skills and compare their progress with friends or other players globally.

6. Additional Game Modes: Introducing new game modes beyond the standard gameplay can add variety and cater to different player preferences. For example, a time trial mode where players compete to complete levels in the fastest time or an endless mode where players aim for the highest score without any level restrictions.

7. Cross-Platform Availability: Expanding the game's availability to different platforms such as mobile devices, desktops, and gaming consoles can greatly increase its reach and potential player base. This allows players to enjoy the game on their preferred devices.

8. Integration of Emerging Technologies: Exploring the integration of emerging technologies like virtual reality (VR) or augmented reality (AR) can provide an immersive and unique gaming experience. This could allow players to control the floppy plane using motion controls or experience the game in a 3D environment.

References


YouTube
YouTube:-YouTube has emerged as one of the best ways for game developers to share how their games are made. And more importantly, how you can make them too.
Unity Documentation
Unity documentation is a comprehensive reference guide for Unity, a popular game development platform.
Google
Develop Android games and deploy them to multiple device types on Android, Chrome OS, and Windows by using the Android Game Development Kit and Google Play