Unity Developer Case Study

Objective

Create a Unity game scenario where the player controls a character using a joystick to move and defeat incoming enemies. The game should feature a timer, and upon completion of the timer, a game won menu will display the number of enemies defeated. The player can then proceed to the next level. Additionally, ensure that the number of enemies defeated is persistently saved, even when the game is closed and reopened.

Scenario

You are tasked with developing a small game in Unity with the following requirements:

1. Character Movement and Enemy Interaction:
   - Implement joystick controls for the character's movement.
   - Enemies should approach the player, and the player must defeat them.

2. Timer and Game End Conditions:
   - Add a timer to the game.
   - When the timer runs out, display a "Game Won" menu showing the number of enemies defeated.
   - Provide an option to proceed to the next level from the "Game Won" menu.

3. Persistent Data:
   - Ensure that the number of enemies defeated is saved and does not reset when the game is closed and reopened.

4. Levels:
   - Develop a total of three distinct levels, each with progressively increasing difficulty and enemy waves.

Assets

- Enemy and player art will be provided.

- Remaining environment and UI elements can use placeholders, such as geometric shapes.

Evaluation Criteria

The following features and patterns will be examined in your implementation:

1. State Machine Pattern:

- Demonstrate knowledge and implementation of the State Machine Pattern for managing game states (e.g., playing, game won, transitioning to the next level).

2. Level System:

- Show understanding of setting up a level system, including transitioning between levels and progressively increasing difficulty.

3. Animator:

- Evaluate the quality of animations and transitions using the Animator component.

4. NavMesh/AI Coding:

- Assess the quality of enemy AI and navigation using NavMesh.

5. Joystick Input Integration:

- Check the smooth integration of joystick input for character control.

6. Object Pooling:

- Utilize object pooling for efficient bullet creation and management.

7. Data Persistence:

- Implement data saving and loading techniques to ensure persistent game data.

8. UI:

- Design a user-friendly and visually appealing UI, including the game won menu.

9. Tweening:

- Use tweening libraries for smooth transitions and animations.

10. Particle Effects:

- Integrate particle effects for enhanced visual feedback during gameplay.

11. Optimization:

- Demonstrate techniques and best practices for optimizing game performance, particularly in handling large enemy waves.

Project Submission

- The project should be shared via GitHub, using the email address muhammet.emin.arday@boombit.com.
- Additionally, provide an APK output of the game attached to the email.

Story Outline

The player controls a hero in a futuristic setting using a virtual joystick. Enemies swarm from all directions in waves, and the hero must eliminate them using various weapons. A timer counts down from 3 minutes, and the player must survive and defeat as many enemies as possible before time runs out.

At the end of the countdown, a "Game Won" screen appears, displaying the total number of enemies defeated. The player can then choose to advance to the next, more challenging level. The game tracks the player's progress, ensuring that their achievements are saved and persist between game sessions.

Each of the three levels introduces new challenges and larger waves of enemies, testing the player's skills and the game's performance optimizations.

By incorporating these features and demonstrating proficiency in the listed evaluation criteria, you will create a robust and engaging Unity game.