Document Design Rationale and Gameplay Impact

Tim Peng

Design and event flow of the system

In this game system, the primary mechanic revolves around collecting items and using those items to trigger gameplay changes, such as gaining temporary boosts and unlocking the level completion objective. The system is driven by Unity's event system, using custom UnityEvent triggers that are invoked whenever a player collects an item, such as a coin or a key.

The key components of the system are as follows:

1. **Inventory system**: The inventory keeps track of the collected items (Gold Keys and Silver Coins) and updates the UI accordingly. As the player collects a specific number of items (5 Gold Keys AND 5 Silver Coins), they will receive a temporary power-up (speed and jump boost for 10 seconds). The inventory is updated dynamically when the player picks up items, with feedback provided via UI text in the bottom left corner.

2. **Player power-up system**: When the player collects a certain number of Gold Keys or Silver Coins, it triggers an event in the system. These events then activate the speed and jump boosts for the player. The boost lasts for 10 seconds, and the player's movement speed and jump height increase during that time, providing more dynamic gameplay and encouraging players to collect items strategically.

3. **Level completion objective**: After the player collects 10 coins or keys, a green beam appears at a random location on the map (in the mountains). The player must touch the beam to complete the level. The appearance of the beam is triggered by the inventory system reaching the threshold (10 items), and its interaction with the player is simplified through a trigger collider.

How events improve modularity and gameplay responsiveness

Events are important to the system's modularity and responsiveness. By using Unity's UnityEvent and custom C# events, each component of the system can function independently while still interacting with one another when necessary. For example:

1. Inventory updates triggers UI changes but are separated from the speed and jump boost system, allowing easy adjustments without changing other game mechanics.

2. Power-up events trigger changes to the player's speed and jump height, but the event can easily be changed to modify other aspects of the player (e.g., speed, jump, etc.) without affecting the inventory system.

3. Level completion is triggered separately by the inventory system, which checks if the required number of items have been collected and then spawns in a green beam in the game world.

This separation of concerns makes the system easier to maintain and expand, allowing me to add new items, power-ups, or other event-driven features with minimal disruption to the existing gameplay mechanics. Furthermore, by managing these interactions through events, the game becomes highly responsive to player actions, allowing the system to react to real-time changes (e.g., picking up an item instantly triggering an inventory update and power-up).

Challenges and solutions during development

1. Inventory system issues:

One of the major challenges was getting the inventory system to track the items properly and reflect the correct counts in the UI. The initial implementation required trial and error to figure out how to correctly update the item count and sync it with the UI. Eventually I was successful after looking at how the teacher implements his own inventory system through his Github and taking a few notes.

2. Collision issues with the cylinder beam:

The green cylinder beam, which appears when 10 items are collected, needed to be interactive with the player. However, there was an issue with its collider, where the beam didn't trigger the player's interaction unless it was set to "Is Trigger." After adjusting the collider to “Is Trigger” in the Inspector, the beam started interacting correctly with the player.

3. Power-up system issues:

The power-up system, which increases the player’s speed and jump height for 10 seconds after collecting 5 Gold Keys or 5 Silver Coins, posed a challenge in terms of proper functionality. Initially, the boost didn’t activate or disappear after 10 seconds. To fix this, the PlayerBoostManager script was moved to the player’s GameObject (the character capsule), rather than having a separate GameObject manage the boost. This allowed for better control over the player’s properties and made the boost system more reliable.

Conclusion

This event-driven system has greatly improved the gameplay by allowing items to quickly change the player's abilities and progression through the level. The use of Unity’s event system allowed for an easily expandable and flexible design where different game mechanics (inventory management, power-ups, level completion) are loosely tied yet still interact with each other in a fluid and responsive manner.

Challenges such as implementing a working inventory system, fixing collision issues with the beam, and ensuring the power-up boosts worked correctly were addressed through trial and error and by adjusting component placements, colliders, and script management. These improvements have resulted in a functional and engaging system that is both flexible and expandable in the future.