Document Design Rationale and Gameplay Impact

Design and Event Flow of the System

The system handles picking up items, storing them in an inventory, dropping them, and recycling them using an event-driven structure. Events make sure everything updates dynamically without constant checking.

When the player picks up an item, ObjectPickup detects it and adds it to the inventory. This triggers an InventoryUpdated event, which updates the UI automatically. When the player drops an item, it is removed from the inventory, and the same event updates the UI.

For recycling, when a can enters the RecyclingBin, it gets destroyed, and GameManager tracks how many have been recycled. Once all cans are recycled, a message appears. Events make this efficient by only updating when necessary instead of running checks every frame.

How Events Improve Modularity and Gameplay Responsiveness

Events keep different parts of the system separate but connected. The UI, inventory, and recycling mechanics don't depend on each other directly but communicate through events. This makes it easy to add new features without rewriting code.

Instead of constantly checking for updates, events trigger changes only when needed. This improves performance and keeps the system flexible. If a new item type were added, it could use the same event system without modifying the core inventory logic.

Challenges and Solutions During Development

One issue was the cans not dropping properly. At first, nothing happened, and no debug logs appeared. The problem was that the inventory wasn't being referenced correctly, so it always thought it was empty. Fixing this allowed the player to drop items again. The drop position was also incorrect, so it was adjusted to spawn cans in front of the player.

A major issue was the UI not updating after picking up or dropping items. This was fixed by making sure the UI subscribed to InventoryUpdated so it refreshed automatically. Another problem was dropped items cloning instead of reusing the original object, which was fixed by modifying DropObject() to reactivate the same object instead of instantiating a new one.

Another problem was objects falling through the ground, which was fixed by properly resetting their position and Rigidbody settings.

Overall, the event system ensures smooth gameplay by keeping updates efficient and allowing easy future expansion.