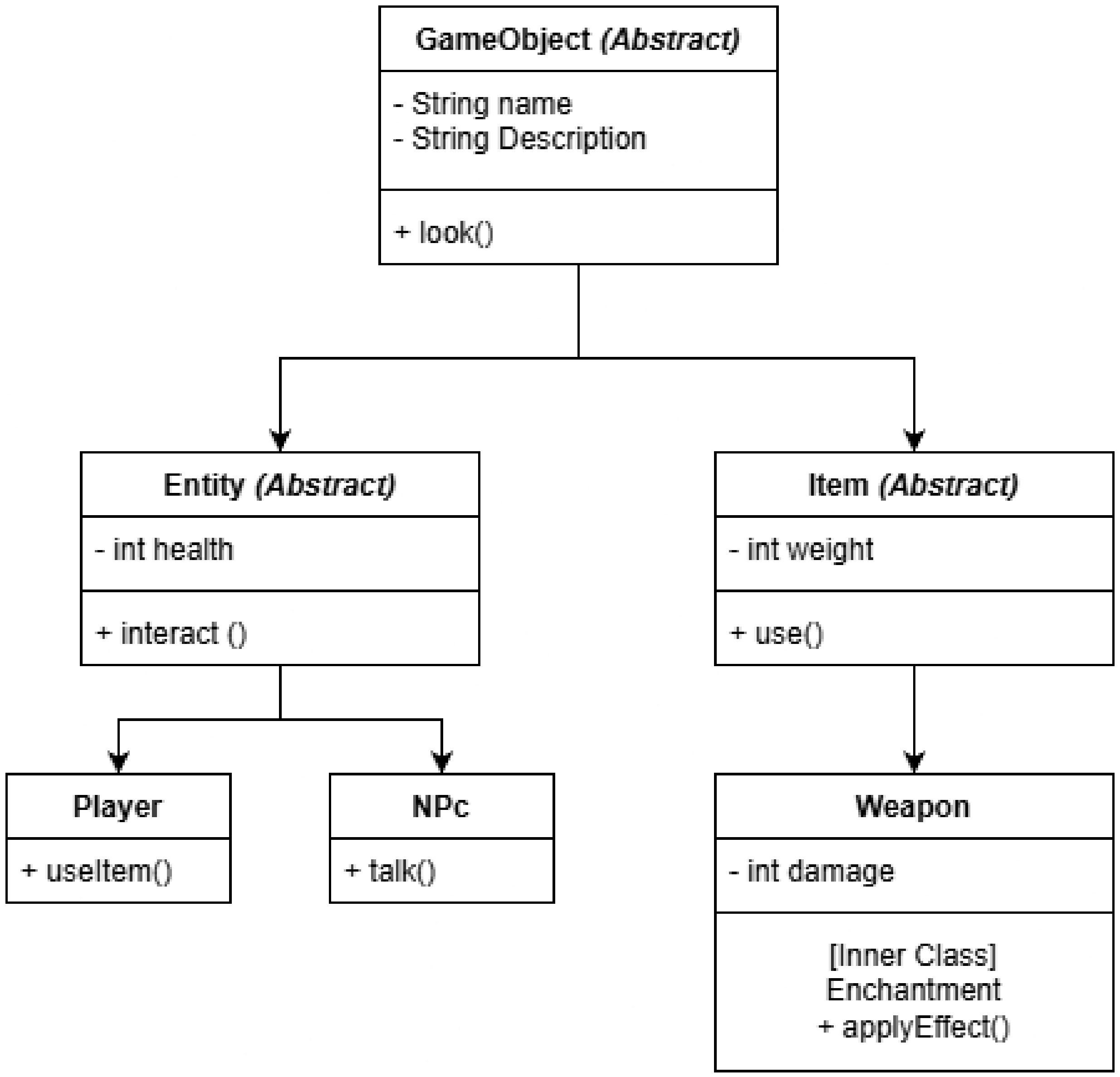
Structure Design

UML Design



Reasoning and Thought Process

Hierarchical Foundation

I started with GameObject as the “Grandparent.” In professional game design, you want a single point of truth. If you later decide that every object in the game should have an unique ID number, you only have to add it to this one class.

Abstract Separation

I split the world into Entity and Item. This keeps code clean, you don’t want a “Sword” inhering code for “Dialogue”, which would happen if you didn’t separate them.

The Power of Override

In this design, the Player and NPs will both override the interact() method. A Player might interact by opening a menu, while an NPC interacts by triggering a text box.

The Choice of Inner Class

I placed Enchantment inside Weapon. My reasoning is scope. An enchantment in this game doesn't exist on its own. It modifies the weapon's properties. By nesting, we indicate that the enchantment’s lifecycle is tied directly to the weapon it belongs to.

Scalability

This design allows for easy expansion. To add a “Consumable” class, you would simply branch it off from Item, Inheriting the use() method automatically.