REQ4 – BUYING WEAPONS

The diagram models the buying process required by REQ4: the Farmer may purchase weapons directly from Merchant NPCs (Sellen or Kale). Weapons (Broadsword, DragonslayerGreatsword, Katana) extend the engine's IntrinsicWeapon for combat stats and implement the Sellable interface to encapsulate buying behavior. Merchants (Sellen, Kale) offer BuyActions conditionally based on the player's runes, injecting different constructor parameters to trigger merchant-specific side effects.

Sellable Interface for Cohesion & DRY

The sellable interface is a minimal interface with getSellPrice(), onSell(Player,GameMap), and getName(). This encapsulates all purchase logic and side effects within each weapon class, so BuyAction depend only on Sellable. This would allow future implementation of Sellable items that aren't weapons.

Weapon Classes

Each weapon class extends IntrinsicWeapon (reusing engine combat mechanics) and implements Sellable. Each class also defines merchant-specific behaviour which is passed in via constructor parameters. This leaves the weapon classes open for extension. New weapons or side effects can be added by creating new classes, without altering existing code (Open/Closed Principle).

Interface Segregation

NPC classes only reference the Sellable interface. Any new sellable item can be offered without modifying merchant logic.

Splitting combat (IntrinsicWeapon) from purchase (Sellable) respects Interface Segregation. Every interface has one clear responsibility.

Trade-Offs & Limitations

- **Duplication in NPC Offerings:** Both Sellen and Kale list their buyable items manually; a shared builder under the merchant could reduce boilerplate. However this would just move the complexity for side effects rather than removing it.
- **Per-Weapon Side Effects:** Housing spawn logic and multiple attribute changes in each class increases class size, but ensures each weapon's unique behaviour stays local and specific to that weapon.

Principles Applied

- **Single Responsibility:** Sellable items handle only their purchase effects.
- **Open/Closed:** New weapons or merchants can be added by subclassing and implementing interfaces, without modifying core engine or existing classes.
- **Low Coupling:** NPCs and BuyAction depend only on Sellable and Action, weapon classes depend on engine abstractions.
- **High Cohesion:** Each class has one clear purpose, combat, purchase effect, or merchant behaviour.