Bock Jennifer, Campbell Anna Maria, Hoelfont Paul, Piterkina Daria, Reitterer Johannes

Refactoring (Testing) - README

Test Strategy

The language used was C# with NUnit. The tests were developed to ensure that the current implementation of Gilded Rose behaves correctly according to the provided “Gilded Rose Specification”.

Test cases were grouped by an items “type” (with its unique behavior):

- Standard Items
- Special Items
    - Aged Brie
    - Sulfuras
    - Backstage Pass

Each test executes only one daily update and checks for changes to Quality and SellIn.

Test Coverage

Standard Items

- Quality degrades by 1 when SellIn > 0
- Quality degrades by 2 when SellIn <= 0
- Quality does not go below 0
- SellIn decreases by 1 daily

Aged Brie

- Quality increases by 1 daily when SellIn > 0
- Quality increases by 2 daily when SellIn <= 0
- Quality doesn’t exceed 50
- SellIn decreases by 1 daily

Sulfuras

- Quality is constant at 80
- SellIn doesn’t change

Backstage Passes

- Quality increases by 1 when SellIn > 10
- Quality increases by 2 when 6 <= SellIn <= 10
- Quality increases by 3 when 1 <= SellIn <= 5
- Quality drops to 0 when SellIn = 0
- Quality doesn’t exceed 50
- SellIn decreases by 1 daily

Refactoring Decisions

The goal of the refactoring was to improve readability and extensibility by introducing separate classes for each type of item, while not adjusting the already implemented Item class. This was done by introducing a wrapper class (WrappedItem), which takes the existing item and wraps it by adding the UpdateQuality() function through an interface. Since the WrappedItem doesn't have to be instantiated directly, it was defined as abstract. Each child class (e.g. AgredBrie, BackstagePass, …) implements their own logic of the UpdateQuality() function. A factory method GetItem(Item item) was implemented in the existing GildedRose class, which maps the different item names to each child class. When looping over the Items collection, an item's implemented UpdatedQuality() function is called, utilizing Polymorphism.