Caso de Negocios: Cafeteria

Key components

- Coffee Interface: Defines the basic structure for any coffee.
- BasicCoffee Class: A simple implementation of the Coffee interface, representing a basic coffee with no extras.
- CoffeeDecorator Class: An abstract class that implements the - Coffee interface and holds a reference to another Coffee object. This class serves as the base for all specific decorators (like Milk, Sugar, etc.).
- Concrete Decorators: These are classes like `MilkDecorator`, `SugarDecorator`, and `WhippedCreamDecorator` that extend the `CoffeeDecorator` and add extra functionality.

This abstract class is the heart of the decorator pattern. It implements the `Coffee` interface and contains a reference (`decoratedCoffee`) to another `Coffee` object.

- **Constructor**: The constructor takes a `Coffee` object as an argument, which it stores in `decoratedCoffee`. This allows the decorator to "wrap" around another coffee object.
- **getDescription()** and **getCost()**: These methods delegate to the `decoratedCoffee` object, calling its `getDescription()` and `getCost()` methods. The concrete decorators will override these methods to add their specific behavior.

The Decorators

- **Constructor**: The `MilkDecorator` constructor takes a `Coffee` object and passes it to the `CoffeeDecorator` constructor. This effectively "wraps" the existing coffee object with a layer of milk.
- **getDescription()**: This method first calls `decoratedCoffee.getDescription()` to get the current description (e.g., "Basic Coffee") and then adds `", Milk"` to it, resulting in something like "Basic Coffee, Milk".
- **getCost()**: Similarly, this method calls `decoratedCoffee.getCost()` to get the current cost, then adds `$0.50` for the milk.

Results

```
Basic Coffee Cost: $2.0
Basic Coffee, Milk Cost: $2.5
Basic Coffee, Milk, Sugar Cost: $2.7
Basic Coffee, Milk, Sugar, Whipped Cream Cost: $3.4000000000000004
```