Soda Machine Console App

Contents

[Objective 1](#_Toc19978172)

[Observations 1](#_Toc19978173)

[About the Solution Stack 3](#_Toc19978174)

[Assumptions 3](#_Toc19978175)

[Flow implemented 3](#_Toc19978176)

[Unit test Project 4](#_Toc19978177)

[Please let me know what you think 5](#_Toc19978178)

# Objective

A piece of code which is analogous to a soda machine application functionality.

Task - Make the existing code beautiful.

# Observations

According to Object Oriented Programming concepts, following is required to be corrected.

1. Hardcoding

For ex - coke.Name == csoda && money > 19 && coke.Nr > 0

This line is too much dependent on the value of money – 19. Tomorrow if this value changes, and if it were an enterprise application too much changes across layers will be require.

1. Tackle Wrong inputs – Removal of fragility is required.

For ex – user enters something other than insert 10 (10 being an integer value). Parsing 10, and parsing wrong user input breaks the code

1. Too many switch cases which impact readability. For a new item introduced in inventory, new switch cases will be required to be added in code.
2. Too difficult testing.
3. Heavy dependency on Soda class

Proposed Solution

1. To remove hardcoding, appropriate usage of classes and collections implemented. Fully testable code. Ready tests for business layer.
2. Inventory Singleton class implemented to keep the logic for input, output coherent and in sync with live soda machine scenario.
3. Business functionalities related to SodaMachine operations reside in BillingMachineOperations.cs. Switch cases dependency on items removed through usage of collections.
4. Removal of wrong inputs removed by having inputs as button number press and through checks implemented at class Beverage and filtering for any wrong input in front layer – SodaMachine.cs
5. Removed dependencies through usage of interfaces.

# About the Solution Stack

**Console application** - .NET Framework 4.6.1

**Libraries** – System, System.Configuration, System.Collection.Generic

**Error Logging** – On to console application window.

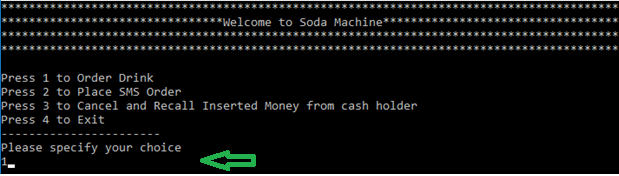
**Initializing Dependencies for tests**: NSubstitute (Nuget package)

**Dependency for ready objects creation**: Unity IOC Container

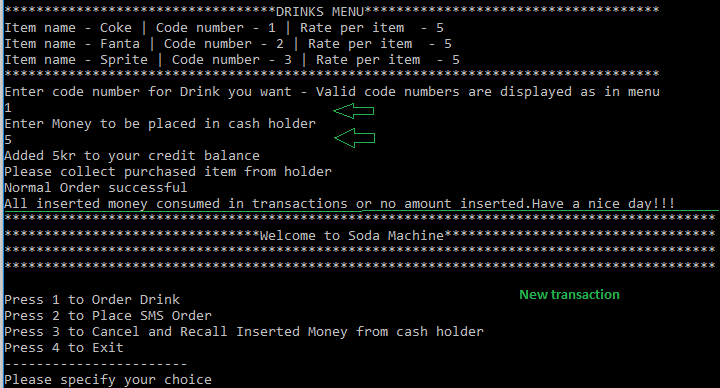
# Assumptions

1. There remains only one inventory integrated to one soda machine. Addition or removal can be achieved for same items or new items.

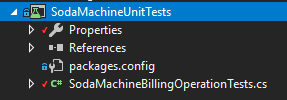
# Flow implemented

Welcome screen

User input choice and ordering flow

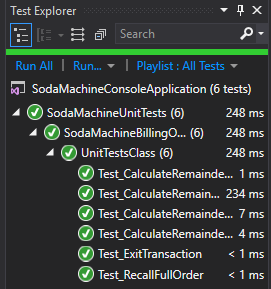


# Unit test Project



Test class contains details of all unit tests created for the project. SodaMachineBillingOperationTests class file tests for all possible scenarios and can be extended for new tests.

**Present state of unit tests-**



**Note – All dependencies for the project is already added in the solution**

# Please let me know what you think

Thanks!