Vending Machine Design Document

Date: 02/18/2024

Author: Garrett Engelder

Table of Contents

Overview

The Vending Machine program in Java is designed to simulate the functionality of a vending machine, allowing users to input cash and make purchases by selecting products through a keypad.

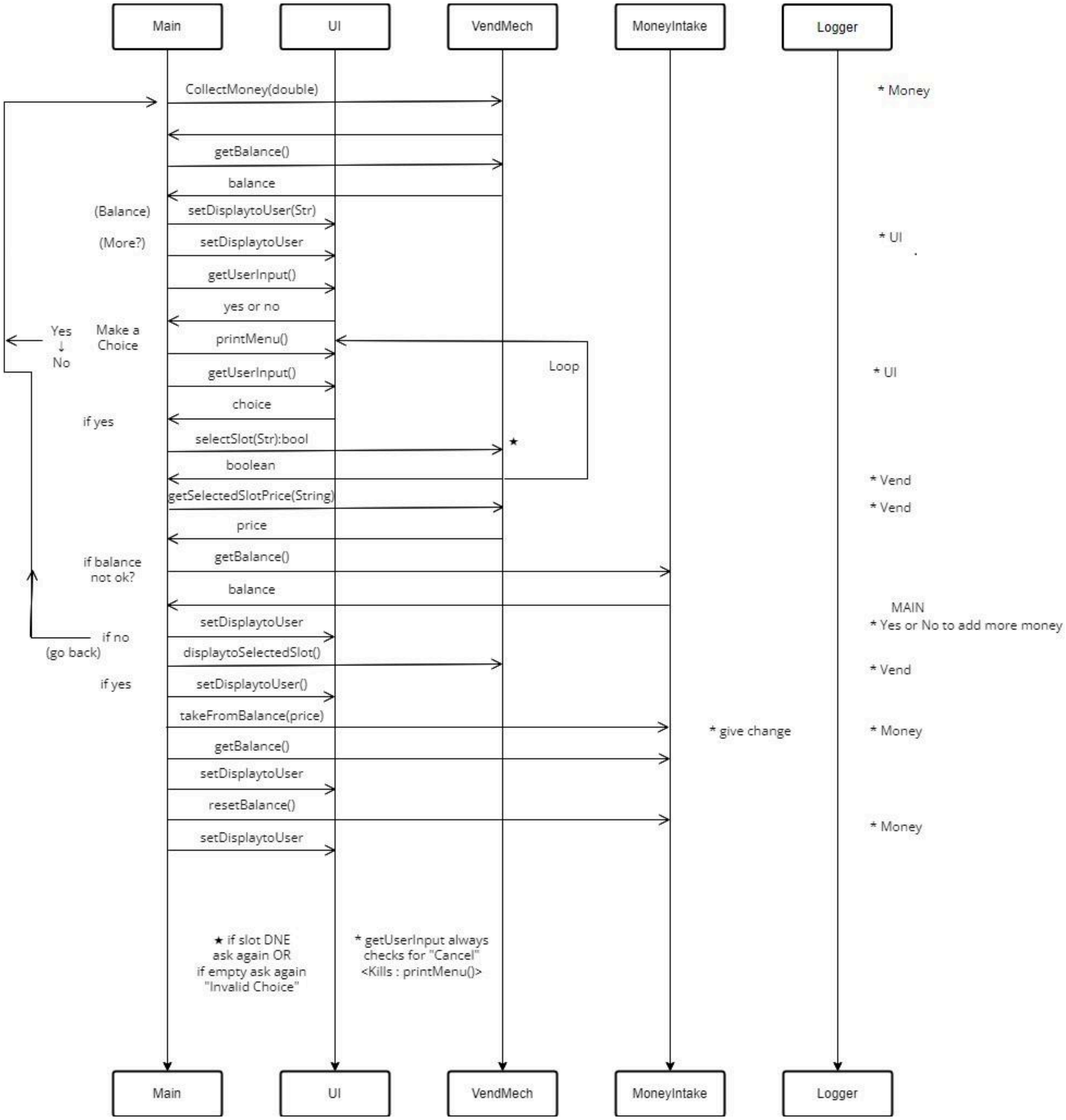
Program Flow

1. **Initialization:**
    - Initialize the program.
    - Display to the user “Insert Cash”.
2. **Flow:**
    - User inputs cash.
    - Display “Make a selection.”
    - Input slot ID selection
    - Run validation checks on selection
    - Dispense product OR display validation check flag
    - Return total balance to user OR allow for new slot ID selection
3. **Program Completion Checks:**
    - **If the player selects a slotID that is of greater value than total balance in machine:**
        - Display “Insufficient Funds”
        - Allow for user to insert cash
    - **If the player selects an invalid slotID:**
        - Display “Invalid ID”
        - Allow for user to select a different slot
    - **If the user selects a product out of stock:**
        - Display “Out of Stock”
        - Allow for user to make a different selection

Program Flow Cont.

4. **Program Completion:**
   - **If the user selects a valid slotID, the product is in stock, and total balance in machine is greater than price of product:**
     - Take price of product from user balance in vending machine
     - Return remaining balance to user
     - Dispense product to user
     - Return to the original state of the program displaying “Insert Cash”
5. **Restart Option:**
   - The user can input “cancel” at any time to retrieve any money they had inserted into the machine and return to the original state of the program displaying “Insert Cash”

Sequence Diagram



Program Structure

Main Program:

- Initializes the controller.
- Begins program loop in thread.

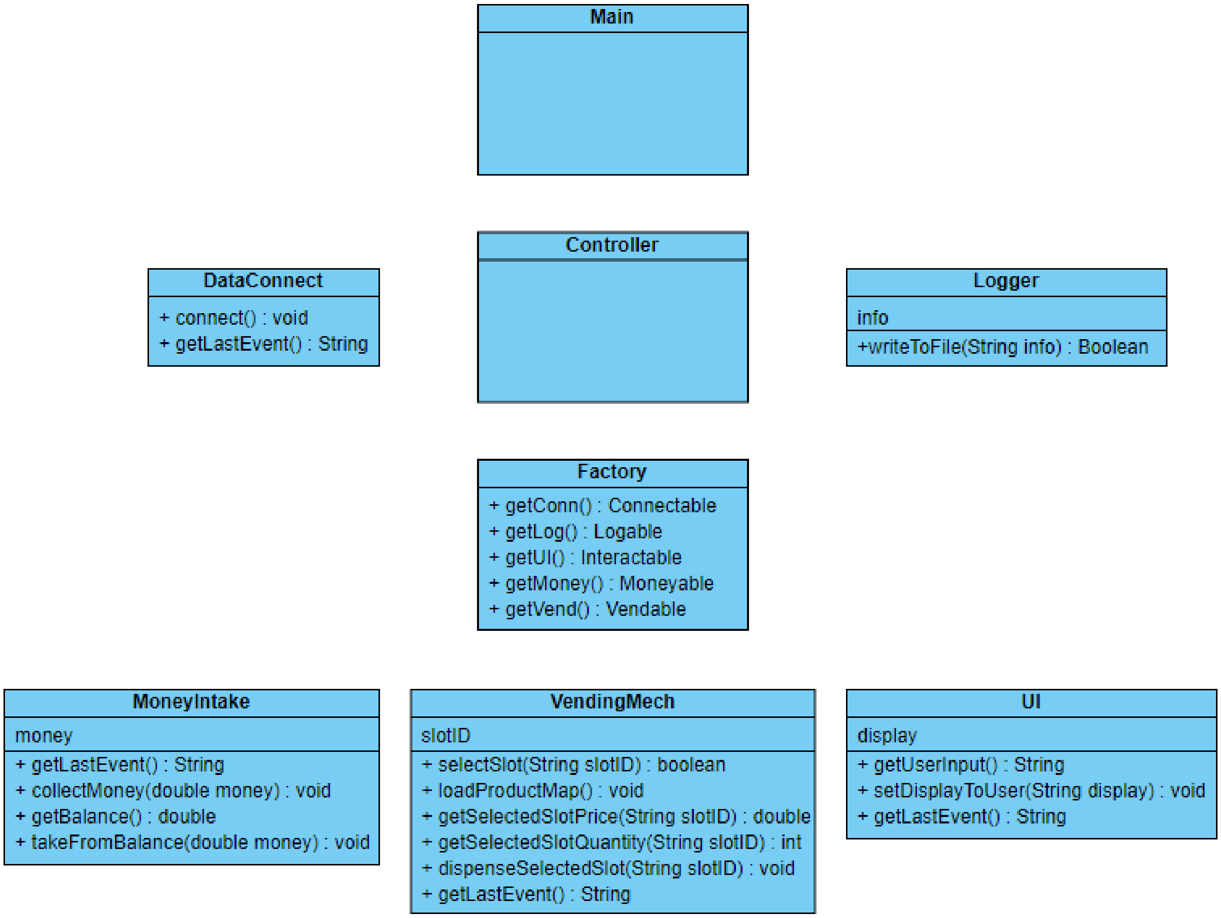
Program Functions:

- getLastEvent() - return info set by other methods
- dispenseSelectedSlot(String slotID) - dispense product selected by user inputted slotID
- getSelectedSlotQuantity(String slotID) - return selected slot quantity
- getSelectedSlotPrice(String slotID) - return selected slot price
- selectSlot(String slotID) - validate user input is a valid slot ID
- VendingMech() - calls loadProductMap method upon creation of class
- setDisplayToUser(String display) - displays local variable 'display' information to user on vending machine display
- getUserInput() - initializes and calls a scanner to retrieve user input
- Slot(double price, int quantity, String productID) - receive created slot information to allow for reading or writing of custom Slot
- stockToFull() - change quantity of slot to full (Not yet implemented)
- stockToN(int n) - change quantity of slot to a specific number (Not yet implemented)
- getPrice() - return price of selected Slot
- getQuantity() - return quantity of selected Slot
- getID() - return quantity of selected Slot (Not yet implemented)
- setPrice(double amount) - change the price of a slot to a specific number (Not yet implemented)

Program Structure Cont.

- collectMoney(double money) - add user inputted cash value to total balance in vending machine
- getBalance() - return total balance in vending machine
- takeFromBalance(double money) - subtract an amount from total balance in vending machine
- writeToFile(String info) - write info to local file log.txt
- Factory() - allow the sub-classes to choose the type of objects to create
- getUI() - return class Ui
- getMoney() - return class Money
- getLog() - return class Log
- getVend() - return class Vend
- getConn() - return class Conn
- connect() - connect to MYSQL database

Class Diagram (W.I.P.)



Main

Controller

DataConnect

+ connect() : void
+ getLastEvent() : String

Logger

info

+writeToFile(String info) : Boolean

Factory

+ getConn() : Connectable
+ getLog() : Logable
+ getUI() : Interactable
+ getMoney() : Moneyable
+ getVend() : Vendable

MoneyIntake

money

+ getLastEvent() : String
+ collectMoney(double money) : void
+ getBalance() : double
+ takeFromBalance(double money) : void

VendingMech

slotID

+ selectSlot(String slotID) : boolean
+ loadProductMap() : void
+ getSelectedSlotPrice(String slotID) : double
+ getSelectedSlotQuantity(String slotID) : int
+ dispenseSelectedSlot(String slotID) : void
+ getLastEvent() : String

UI

display

+ getUserInput() : String
+ setDisplayToUser(String display) : void
+ getLastEvent() : String

Development Challenges and Current Issues

Future Enhancements

Conclusion

The Vending Machine program provides a foundational framework for creating a simulated vending machine experience.