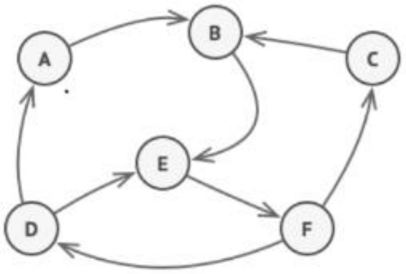
Vending Machine

State Design Pattern

State is a behavioral design pattern that lets an object alter its behavior when its internal state changes. It appears as if the object changed its class.



Finite-State Machine.

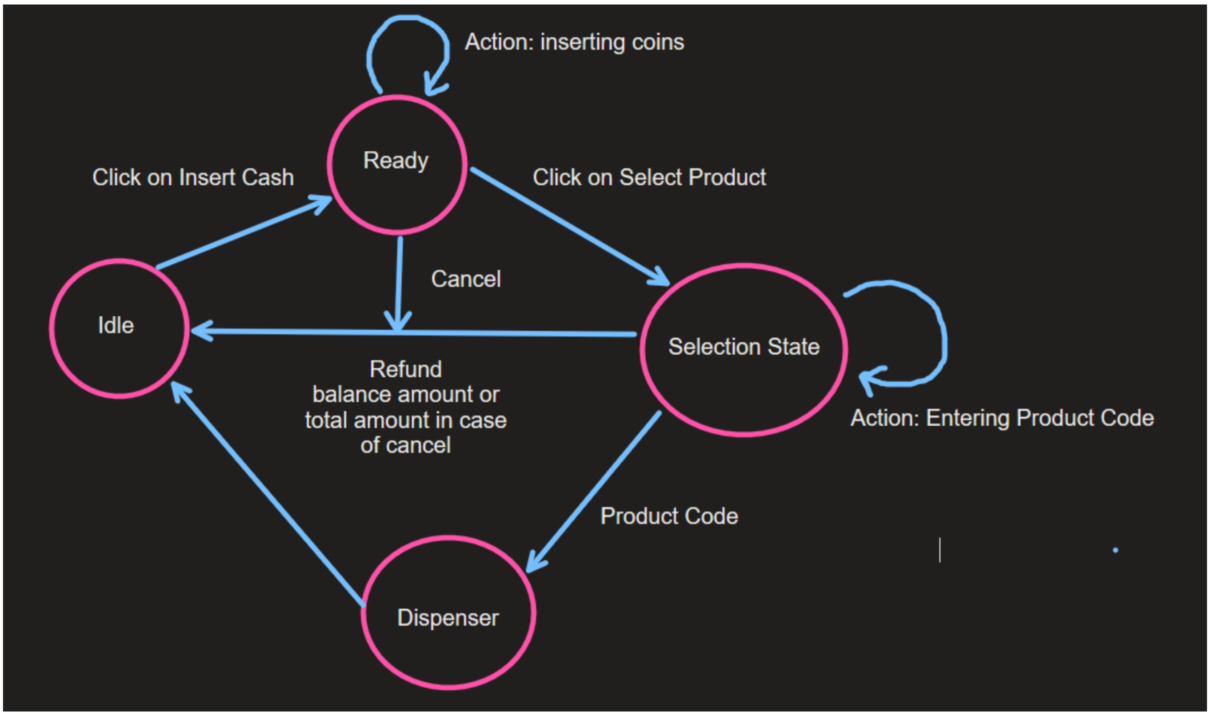
The main idea is that at any given moment there's a finite number of states which a program can be in. Within any unique State, the program behaves differently, and the program can be switched from one State to another instantaneously. However, depending on a current state, the program may or may not switch to certain other states. These switching rules, called transitions, are also finite and predetermined.

Design Vending Machine

Requirements:

- Should be able to click on insert button cash.
- Able to Insert Cash.
- Should be able to do product selection.
- Able to Select some product – 123
- Checker - Check if the amount is more than the product price
- I click on cancel; entire amount needs to be refunded.
- It will dispense the product
- Dispense the change in the cash dispenser

State Diagram:



Vending Machine

- List<Coins> coins
- Inventory
- State

State

- Idle
- Ready
- Selection
- Dispense

Product

- Name and Type
- Price

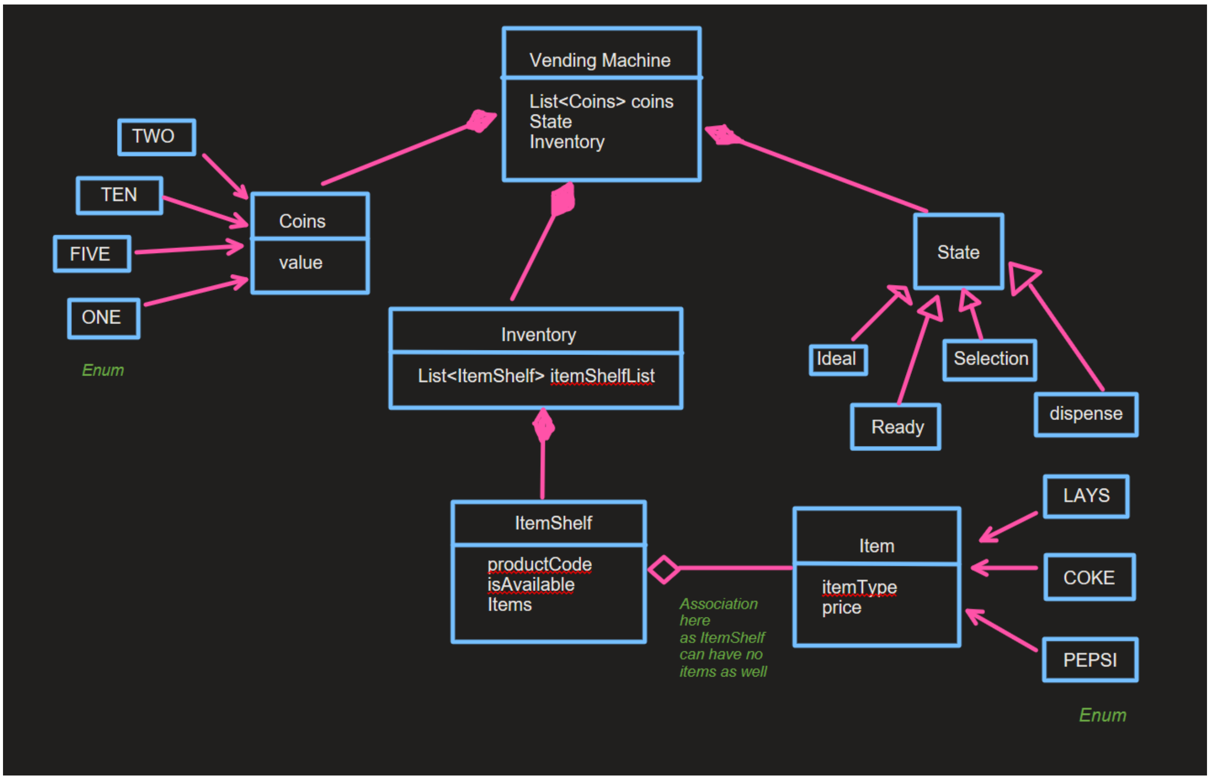
ItemShelf

- Product Code
- isAvailable
- Items

Inventory

- List<ItemShelf> itemShelf

Class Diagram


Vending Machine
List<Coins> coins
State
Inventory
TWO
TEN
FIVE
ONE
Coins
value
Enum
Inventory
List<ItemShelf> itemShelfList
State
Ideal
Selection
Ready
dispense
ItemShelf
productCode
isAvailable
Items
Item
itemType
price
LAYS
COKE
PEPSI
Association here as ItemShelf can have no items as well
Enum