Kata Instructions

- All kata requirements do not need to be coded in order for the kata to be submitted, however requirements implemented should be done so using best practices. **This is a showcase of skill, not speed.**
- When submitting, please let us know what kata requirements were implemented.
- Spend no more than **two hours** on each kata.
- All code should be **test-driven** and **committed**.
- See **below** for the instructions for the **Kata** to be completed and a link to the source of the Kata.

Getting Started

1. Save the java_audition.bundle to your system.

   Go to http://git-scm.com/blog/2010/03/10/bundles.html for more information on git bundles. These files are compressed git repositories of empty project shells that you will be working within. The Java project uses maven.

2. After the files are saved, extract them with git with the following command(s):
   `git clone java_audition.bundle -b master [candidates_name]_java_audition`
   Example: `git clone java_audition.bundle -b master danmonroe_java_audition`

3. Complete the katas. Commit your changes as you develop.

4. When completed, create a new bundle with the implemented kata:
   `git bundle create [candidates_name]_java_audition.bundle master`
   Example: `git bundle create danmonroe_java_audition.bundle master`

5. Verify your bundle before submitting:
   `git bundle verify [candidates_name] _java_audition.bundle`
   Example: `git bundle verify danmonroe_java_audition.bundle`

Vending Machine Kata (https://github.com/guyroyse/vending-machine-kata)

In this exercise, you will build the brains of a vending machine. It will accept money, make change, maintain inventory, and dispense products. All the things that you might expect a vending machine to accomplish.

The point of this kata to to provide a larger than trivial exercise that can be used to practice TDD. A significant portion of the effort will be in determining what tests should be written and, more importantly, written next.

Features

Accept Coins

As a vendor
I want a vending machine that accepts coins
So that I can collect money from the customer

The vending machine will accept valid coins (nickels, dimes, and quarters) and reject invalid ones (pennies). When a valid coin is inserted the amount of the coin will be added to the current amount and the display will be updated. When there are, no coins inserted, the machine displays INSERT COIN. Rejected coins are placed in the coin return.

NOTE: The temptation here will be to create Coin objects that know their value. However, this is not how a real vending machine works. Instead, it identifies coins by their weight and size and then assigns a value to what was inserted. You will need to do something similar. This can be simulated using strings, constants, enums, symbols, or something of that nature.

Select Product

As a vendor
I want customers to select products
So that I can give them an incentive to put money in the machine

There are three products: cola for $1.00, chips for $0.50, and candy for $0.65. When the respective button is pressed and enough money has been inserted, the product is dispensed and the machine displays THANK YOU. If the display is checked again, it will display INSERT COIN and the current amount will be set to $0.00. If there is not enough money inserted then the machine displays PRICE and the price of the item and subsequent checks of the display will display either INSERT COIN or the current amount as appropriate.

Make Change

As a vendor
I want customers to receive correct change
So that they will use the vending machine again

When a product is selected that costs less than the amount of money in the machine, then the remaining amount is placed in the coin return.

Return Coins

As a customer
I want to have my money returned
So that I can change my mind about buying stuff from the vending machine

When the return coins' button is pressed, the money the customer has placed in the machine is returned and the display shows INSERT COIN.

Sold Out

As a customer
I want to be told when the item I have selected is not available
So that I can select another item

When the item selected by the customer is out of stock, the machine displays SOLD OUT. If the display is checked again, it will display the amount of money remaining in the machine or INSERT COIN if there is no money in the machine.

Exact Change Only

As a customer
I want to be told when exact change is required
So that I can determine if I can buy something with the money I have before inserting it

When the machine is not able to make change with the money in the machine for any of the items that it sells, it will display EXACT CHANGE ONLY instead of INSERT COIN.