Point of Sale (PoS) Systems ~ Thomas Yamakaitis

## About

The following documentation details the usage of multiple versions of the same or similar PoS systems. These are just GUI versions for testing purposes. They are meant to assist developers at the Compuflex Corporation to test screen mapping and management software(s). These application(s) are, simply put, Graphical representations of actual PoS systems, and they mimic common behavior(s) in such systems.

## Versions

- PoS (Java - JavaFX) v1.3 - This is probably one of my best implementations of this project. It was created with Java / JavaFX using Eclipse IDE and Scene Builder 2.0 software.
- PoS (Java - Swing) v2.0 - This is another Java version of the project using a different `flavor`. It was created with Java / Swing / AWT using Eclipse IDE and it's `WindowBuilder` plugin.
- MockPoS (JavaScript) - This implementation was done using web technologies including:
    - JavaScript
    - jQuery v3.2.1
    - Bootstrap v3.3.7
    - Font Awesome v4.7.0
    - HTML5 & CSS3

## Overview

## 1.) Configuration & Settings

These applications have a few settings and configurable options to mimic a variety of behaviors that can occur in actual user interfaces for PoS systems.

### a.) Opening the Settings Window

You can open the settings window by right-clicking the top bar on any version of this application. Here you will see a variety of options.

#### i.) Pop-up Pay Window

Some PoS systems behave differently than others, some have a pop-up window, whereas some might show the whole process in one single window. You can toggle this option in every version using the radio button(s) provided in the Settings window (Refer back to 1a. "Opening the Settings Window").

******[NOTE: FOR JAVA SWING USER(S): There are no radio buttons, just a simple toggle button, this will be changed in the next released version for uniformity.]******

*To manually change this setting: (***ONLY IN JAVAFX AND JAVA SWING VERSIONS***)*

1. Close the program.
2. Open the `.properties` file which has the same name as your `.jar`
3. Change `popUp = false` to `popUp = true` to turn on the pop-up pay window, or vice versa to turn it off.
4. Re-open the program.

#### ii.) User Display Options

Once again, some PoS systems behave differently than others, some display usernames as a button, others as plain text, others might have a dropdown menu, and some might even be images. This application supports each of these behaviors. You can switch between these options using the radio button(s) provided in the Settings window (Refer back to 1a. "Opening the Settings Window").

*To manually change this setting: (***ONLY IN JAVAFX AND JAVA SWING VERSIONS***)*

1. Close the program
2. Open the `.properties` file which has the same name as your `.jar`
3. Change `userDisplay = <insert integer between 0-3 here>`, if you put an integer other than `0-3`, the program will use it's default. Check the `default.properties` file for a reference.
4. Re-open the program.

b.) Properties Files (***ONLY IN JAVAFX AND JAVA SWING VERSIONS***)

Java offers a built-in properties class that maps to a "Key-Value" pair in a related `.properties` file.

For your information, if these file(s) are not in the directory where your `.jar` is located, the file(s) will be automatically generated within the application for your convenience.

When you change settings in the Settings window (Refer back to 1a. "Opening the Settings Window"), if you press `Okay` or `OK`, it will automatically save these settings to the `.properties` file.

When you open the program, it will reference these settings and display the UI based on those settings. The `default.properties` file is just a default file for reference, it is okay to delete it, but the program will automatically generate it if no `.properties` file is found. The main `.properties` file is named based on the `.jar` file, and it is also automatically generated, if deleted.

2.) The Cart

---

The cart stores item(s) that are to be processed into a transaction, it generates a total price based on the individual price(s) of the item(s) in the cart. (i.e. for every 1 item priced at $1.00 in the cart, $1.00 will be added to the total price). You can view the cart by clicking the `Cart` tab at the top left.

a.) Adding an item to the cart

Click the `Add` button at the top left of the window (which is to the left of the `Clear` button, left of the `Remove` button in the JavaScript version) to add an item to the cart, the item will be randomly generated based on hard-coded names and prices in the source code. After clicking `Add`, the item will be displayed in a graphical table, and the value of `Total` at the bottom left of the window will be incremented.

b.) Clearing the cart

Click the `Clear` button at the top left of the window (which is to the right of the `Add` button) to clear all item(s) from the cart. This will clear all item(s) from the cart, cancel the transaction, and set the value of `Total` at the bottom left to `$0.00`, effectively, resetting every value to default.

c.) Removing an item from the cart (***ONLY IN JAVASCRIPT VERSION***)

To remove a specific item from the cart, simply click the checkbox next to that item in the table and it will remove all instances of that item.

******[NOTE: This operation was only done in the JavaScript version for the sake of simplicity in the Java version(s).]******

## 3.) The Transaction Record

The transaction record stores transaction(s) which contain information about a cart that was processed through the system at the Point of Sale. It holds the total price of the cart, the amount of currency tendered, and the amount of change returned as well as the time that the transaction took place and the user who processed it. You can view the transaction record by clicking the `Trans` tab at the top left, to the right of the `Cart` tab. The transaction record has no operation(s) because transactions are not meant to be tampered with, they record one's history of transactions.

### a.) Processing a Transaction

To process a transaction, first you must add items to the cart (Refer back to 2a. "Adding an item to the cart"), then click the `Pay` button located at the bottom right corner.

Depending on your settings, either a pop-up window will appear or fields will appear within the same window (Refer to 3b. "Payment Fields").

Next, you will enter a ***numeric*** value into the `Tendered` text field. The `Tendered` amount must be ***greater than or equal to*** the `Total` amount, which means the `Required` amount must be `$0.00` ***or LESS*** in order to continue.

Once those conditions are met, the `Process` button will be enabled and you can click it to process the transaction. This will store a transaction in the transaction record with the current time and user, as well as the values of the fields (Refer to 3b. "Payment Fields"). This will also clear the cart and reset all fields to default or `$0.00`. This will also hide the `Tendered` and `Required` and show the `Pay` button once again.

### b.) Payment Fields

There are 3 payment fields:

1. `Total` which is shown on opening the program. `Total` is the total sum of (***the quantity * the price***) of each item in the cart.
2. `Tendered` which is shown upon pressing the `Pay` button. `Tendered` is the amount to be tendered from the customer.
3. `Required` which is also shown upon pressing the `Pay` button. `Required` is the value of `Total - Tendered`. (i.e. if `Total = $20.00` and `Tendered = $23.00`, then `Required = (Total - Tendered) = ($20.00 - $23.00) = -$3.00`, therefore `Required = -$3.00` which means the customer is owed `$3.00` in change).