Hardware Rental Store Report & UML

Team: Dieu My Nguyen, Adil Sadik, Yu Li

Design Report

In our rental store project, we implemented a few classes to simulate the store: `Tool`, `Rental` and `Customer`. There are also collection classes which initialize and contain these classes' objects. The design of this simulator implements a Facade pattern in the class `StoreFacade`, to use some of the functions in customer, rental and tool collections to simulate actions such as `process_returns()`, `rent_tools()`, `retrieve_rental_records()` and etc. for the main program to use. The main program acts like a driver that simulates the store over 35 days and prints out a report at the end.

The `Tool` class is used to define a "tool" in the rental store. It is implemented in a polymorphic way: the class `Tool` itself is an abstract class with attributes `name` and `available` to represent whether the tool is rented out. The `Tool` class is inherited by five different tool classes to represent the different categories of tools the store has. These derived subclasses have their own `category` and `price` attributes.

`Tool` objects are aggregated by a class called `ToolCollection` that acts as an inventory. Tools are generated within this class. This class is also in charge of renting and restocking tools with two methods `restock` and `rent`, which decreases or increases the inventory overall. This class also iterates through and prints the available tools with the method `print_tool_list()`.

Customers are defined with the class `Customer`, which contains attributes such as customer's name, the choices for number of tool and night according to their type, and the number of tools rented to keep track of how many tools they currently have in possession and determine whether they can rent in a given day. The number of days and the number of tools that customer will choose based on their type are generated within the `Customer` class with two methods: `choose_num_tools()` and `choose_num_nights()`. The `Customer` classes are implemented in a polymorphic way: three different kinds of customer classes extends the general `Customer` class: `CasualCustomer`, `BusinessCustomer`, `RegularCustomer`.

Customer objects are aggregated with a class called `CustomerCollection`, in which a customer pool is randomly generated. When the store has inventory available, the main program will use the `StoreFacade` class method `rent_tools()` (describe more below) to randomly select a customer who has not rented more than 2 tools. The customer collection uses method `return_tool()` to maintain the correct number that tools a customer have. This class can print a list of customers showing their attributes.

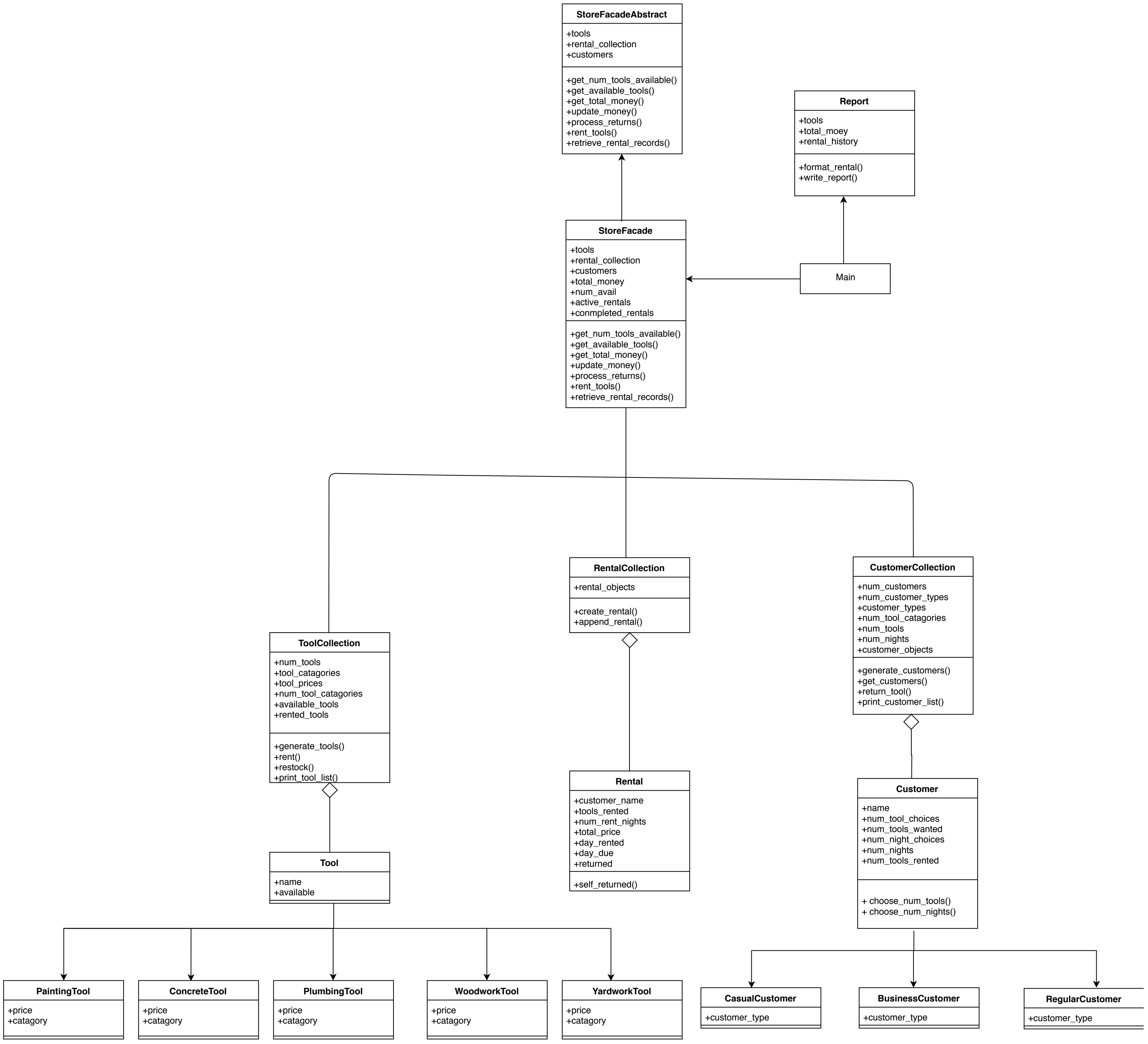
The class `Rental` is used to keep track of the renting order that customers make. It includes the name of the customer who made the rental, number of tools that a rental contains, the date that the rental is made and how long the rental lasts, whether the rental is returned or not.

The class `RentalCollection` aggregates and generates rental order. It keeps a history of all the rentals made with method `append_rental()`.

The class `StoreFacade` is a faade derived from the `StoreFacadeAbstract` interface that aggregates `ToolCollection`, `CustomerCollection`, and `RentalCollection` to simulate a store. The store keeps track of the existing rentals along with the current inventory of the store. As such, when it has zero rentals, there will be 20 tools in its inventory. When it has zero tools in its inventory, it will have multiple rentals that between them account for all 20 tools. The main program uses this faade interface to simulate checking inventory, return processing, tool renting, and sorting completed and active rentals in a given day.

At the end of simulation, the main program uses the class `Report` generates a `Report` object that prints out the current store stock, total money made and the current active rentals and completed rentals.

UML (See attached pdf for more legible version)


StoreFacadeAbstract
+tools
+rental_collection
+customers
+get_num_tools_available()
+get_available_tools()
+get_total_money()
+update_money()
+process_returns()
+rent_tools()
+retrieve_rental_records()
Report
+tools
+total_moey
+rental_history
+format_rental()
+write_report()
StoreFacade
+tools
+rental_collection
+customers
+total_money
+num_avail
+active_rentals
+conmpleted_rentals
+get_num_tools_available()
+get_available_tools()
+get_total_money()
+update_money()
+process_returns()
+rent_tools()
+retrieve_rental_records()
Main
RentalCollection
+rental_objects
+create_rental()
+append_rental()
CustomerCollection
+num_customers
+num_customer_types
+customer_types
+num_tool_catagories
+num_tools
+num_nights
+customer_objects
+generate_customers()
+get_customers()
+return_tool()
+print_customer_list()
ToolCollection
+num_tools
+tool_catagories
+tool_prices
+num_tool_catagories
+available_tools
+rented_tools
+generate_tools()
+rent()
+restock()
+print_tool_list()
Rental
+customer_name
+tools_rented
+num_rent_nights
+total_price
+day_rented
+day_due
+returned
+self_returned()
Customer
+name
+num_tool_choices
+num_tools_wanted
+num_night_choices
+num_nights
+num_tools_rented
+ choose_num_tools()
+ choose_num_nights()
Tool
+name
+available
PaintingTool
+price
+catagory
ConcreteTool
+price
+catagory
PlumbingTool
+price
+catagory
WoodworkTool
+price
+catagory
YardworkTool
+price
+catagory
CasualCustomer
+customer_type
BusinessCustomer
+customer_type
RegularCustomer
+customer_type