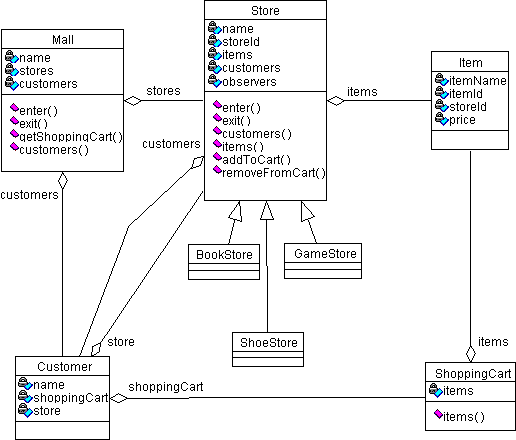
|  |  |
| --- | --- |
|  | Pontificia Universidad JaverianaIngeniería De SistemasIngeniería De SoftwareTaller: Diseño de Software |

In this Laboratory, you will be DESIGNING a simple shopping mall in Java. The mall will provide a soothing shopping experience for your customers, while at the same time allowing you to explore the wonderful world of design patterns!

The class diagram which provides the framework for the shopping mall is as follows:



Here's a brief description of each class:

1. Mall: The mall itself. Either an applet or application. The mall performs the creation of a set of different stores, such as a book store, a shoe store, an electronic games store, etc. The mall greets an arriving customer and allows the customer to shop at the mall.

Some of the things a customer can do at the mall are:

+ Get a list of available stores

+ Get a shopping cart

+ Enter a store

+ Get of list of items available for sale at the store

+ Add items to the shopping cart

+ Proceed to checkout and purchase the items

o Notable Attributes:

+ name - the name of the mall

+ stores - a collection of stores of different types

+ customers - the customers currently in the mall

o Possible Methods:

+ void enter(Customer c) - customer c enters the mall

+ void exit(Customer c) - customer c exits the mall

+ ShoppingCart getShoppingCart - returns an empty shopping cart

+ Enumeration customers() - returns an enumeration of the customers in the mall

+ void checkout(shoppingCart cart) - checkout and purchase the items in the shopping

cart

1. Store

o Abstract superclass for a store

o Notable Attributes:

+ name - the name of the store

+ storeId - unique ID for the store

+ items - items available for sale in the store

+ customers - the customers currently in the store

o Possible Methods:

+ abstract void enter(Customer c) - customer c enters the store

+ abstract void exit(Customer c) - customer c exits the store

+ Enumeration customers() - returns an enumeration of the customers in the store

+ Enumeration items() - returns an enumeration of the items available for sale in the

store

+ abstract void addToCart(shopingCart, item) - add an item to the shopping cart

+ abstract void removeFromCart(shopingCart, item) - remove an item from the shopping cart

1. BookStore

o A possible subclass of Store

1. ShoeStore

o A possible subclass of Store

1. GameStore

o A possible subclass of Store

1. Item

o An item for sale in a store

o Notable Attributes:

+ itemName - the name of the item

+ itemId - unique ID for the item

+ storeId - the ID of the store from which the item came

+ price - the price of the item

1. Customer

o A customer!

o Notable Attributes:

+ name - the name of the customer

+ shoppingCart - the shopping cart being used by the customer

+ store - the store the customer is currently in

1. ShoppingCart

o A shopping cart for the customer

o Notable Attributes:

+ items - items currently in the shopping cart

o Possible Methods:

+ Enumeration items() - returns an enumeration of the items currently in the cart

Note that required accessors and mutators are not listed in the above for the sake of brevity.

## Project Requirements

Use the above framework to REDESIGN the shopping mall.

Your project must have at least one customer, three stores and five items for sale at each store.

Your project must implement the following design patterns:

1. Iterator: This one is easy. Methods that return an Enumeration demonstrate the Iterator pattern.
2. Abstract Factory with Factory Methods: Create all stores using factories.
3. Singleton: Allow only one instance of any factory type you create.
4. Observer: All stores should be observable objects. In particular, a customer should be able to be notified whenever a new item has been added for sale at a store. Demonstrate this pattern by having one or more customers register for such notifications with one or more stores.

NOTE: The methods required to implement this pattern are not specified above!

Feel free to use any other patterns you feel appropriate.

Your DIAGRAMS must also be well documented. In particular, places where design patterns are implemented should be noted.

Finally, a short report, no more than 4 pages in length, should be submitted describing your use of design patterns in your project.

THIS LAB WAS TAKEN FROM: http://userpages.umbc.edu/~tarr/dp/fall00/projects/Project1.html