Final Bookstore Design

Index

Class Excptns

This class holds all the exceptions which can occur in this project.

Constructor Detail

- **Excptns**
  `public Excptns()`

Method Detail

- **invalidDouble**
  `public static double invalidDouble()`
  This Exception handler handles InputMismatchException for any double value. It has a while loop for letting the user enter values till the valid input is entered.
  Returns:
  Double - The Double value as price.
- **invalidInt**
  `public static int invalidInt()`
  This Exception handler handles InputMismatchException for an int value. It has a while loop for letting the user enter values till the valid input is entered.
  Returns:
  Integer - This will be any correct int value.
- **fileNotFoundExc**
  `public static java.lang.String fileNotFoundExc()`
  This Exception handler handles FileNotFoundException for any String name typed. It has a while loop for letting the user enter values till the valid input is entered.
  Returns:
  String - This will be the file name that was successfully found.
- **stringToDouble**
  `public static double stringToDouble(java.lang.String srt)`
  This Exception handler handles when the scanner reads the file. It catches an exception when the value is not double after the string has been parsed.

Parameters:

`srt` - this will hold the value in a string format and will be used to Parse double value

Returns:

double - This will be the successfully parsed double value from String.

- **stringToInt**

  `public static int stringToInt(java.lang.String srt)`

  This Exception handler handles when the scanner reads the file. It catches an exception when the value is not an int after the string has been parsed.

  Parameters:

  `srt` - this will hold the value in a string format and will be used to Parse integer value

  Returns:

  Integer - This will be the successfully parsed int value from String.

- **checkDuplicateMember**

  `public static boolean checkDuplicateMember(int id)`

  This method checks for duplicate members when adding a new member to the member's list.

  Parameters:

  `id` - this method excepts an integer which is compared to other integer values

  Returns:

  Boolean - This will be either true or false if there are duplicate members.

Class Inventory

This class takes care of the entire inventory of products and members in this project.

- ***Constructor Detail***
  - **Inventory**
    `public Inventory()`
- ***Method Detail***
  - **getItems**
    `public java.util.ArrayList<Item> getItems()`
    this method returns the item list
    Returns:
    ArrayList this list will be the list of items
  - **getMember**
    `public java.util.ArrayList<Member> getMember()`
    this method will return a list of the members
    Returns:
    ArrayList the list of members
  - **setItems**
    `public void setItems(java.util.ArrayList<Item> items)`
    this method sets the item list to the list passed in the parameter
    Parameters:
    `items` - an ArrayList of items
  - **setMember**
    `public void setMember(java.util.ArrayList<Member> member)`
    this method sets the member list to the list passed in the parameter
    Parameters:
    `member` - an ArrayList of members
  - **addMember**
    `public void addMember(Member member)`
    this method adds a number to the member list
    Parameters:
    `member` - A member object
  - **addItem**
    `public void addItem(Item item)`
    this method adds an item to the item list
    Parameters:
    `item` - an item object

- **removeItem**

  `public void removeItem()`

  this message removes an item by asking the user, the item ID.
- **displayItems**

  `public void displayItems()`

  this method displays all the items in the item list in a table format.
- **insertionSortItems**

  `public void insertionSortItems()`

  this is a insertion sort method to sort the items according to their type.
- **displayMember**

  `public void displayMember()`

  this method displays all member in the table format.
- **insertionSortMembers**

  `public void insertionSortMembers()`

  this is insertion sort method to sort all the members according to their type.

Class Item

This class has the basic data structure for an item that can be a Book or CD or a DVD.

- ***Constructor Detail***

Item
`public Item(java.lang.String name,double price,int amount)`
Items Constructor without the type. This will be used when a user adds any item manually.
Parameters:
`name` - string value
`price` - double value
`amount` - integer value

Item
`public Item(java.lang.String type,java.lang.String name,double price,int amount)`
Items Constructor. This will be used when a user adds an item from a file.
Parameters:
`type` - string value
`name` - String value
`price` - double value
`amount` - integer value

- ***Method Detail***
  - **getType**
    `public java.lang.String getType()`
    This method gets the item type.
    Returns:
    string value of the type
  - **getName**
    `public java.lang.String getName()`
    this method gets the name of the item
    Returns:
    string type value of name
  - **getPrice**
    `public double getPrice()`
    this method gets the price of the item
    Returns:
    double type value of price

- **getAmount**

  `public int getAmount()`

  this method gets the quantity of the item

  Returns:

  integer type value of the price
- **setType**

  `public void setType(java.lang.String type)`

  this method sets the type of the item

  Parameters:

  `type` - the string type for name of the type of item
- **setName**

  `public void setName(java.lang.String name)`

  this method sets the name of the item

  Parameters:

  `name` - string type
- **setPrice**

  `public void setPrice(double price)`

  this method sets the price of the item

  Parameters:

  `price` - double type
- **setAmount**

  `public void setAmount(int amount)`

  this method sets the quantity of the item

  Parameters:

  `amount` - integer type

Class Member

This class maintains the list of the members which are free and premium and creates new members through files.

- ***Constructor Detail***

Member
`public Member(java.lang.String type,int id)`
constructor with two perimeter
Parameters:
`type` - string
`id` - integer

Member
`public Member(java.lang.String type, int id, boolean hasPaid)`
Constructor with three-parameter
Parameters:
`type` - string
`id` - integer
`hasPaid` - boolean

- ***Method Detail***
  - **getType**
    `public java.lang.String getType()`
    this method gets the type of the member
    Returns:
    string type
  - **getId**
    `public int getId()`
    this method gets the ID of the member
    Returns:
    integer type
  - **isHasPaid**
    `public boolean isHasPaid()`
    this method gets true or false if the member has paid or no.
    Returns:
    boolean type

- **setType**

  `public void setType(java.lang.String type)`

  this method sets the type of the member

  Parameters:

  `type` - string
- **setId**

  `public void setId(int id)`

  this method sets the ID of the member

  Parameters:

  `id` - integer
- **setHasPaid**

  `public void setHasPaid(boolean hasPaid)`

  this method sets true or false if the member has paid or no.

  Parameters:

  `hasPaid` - boolean
- **toString**

  `public java.lang.String toString()`

  Display the member's list.

  Overrides:

  `toString` in class `java.lang.Object`

  Returns:

  string

Class Order

This class takes care of all the purchases done and adds items to the cart and creates an invoice.

- ***Constructor Detail***
  - **Order**

    `public Order()`
- ***Method Detail***
  - **getIdAndQuantityList**

    `public static java.util.ArrayList<int[][]> getIdAndQuantityList()`

    this method gets the original item ID and the quantity requested by the user when an item is added to the cart.

    Returns:

    ArrayList of integer 2 D array
  - **getPaymentMethod**

    `public java.lang.String getPaymentMethod()`

    this method gets the payment method used to checkout.

    Returns:

    string type
  - **setIdAndQuantityList**

    `public static void setIdAndQuantityList(java.util.ArrayList<int[][]> idAndQuantityList)`

    this method sets the original item ID and the quantity requested by the user in a 2D array when an item is added to the cart.

    Parameters:

    `idAndQuantityList` - Array list of 2d Array
  - **setPaymentMethod**

    `public void setPaymentMethod(java.lang.String paymentMethod)`

    this method sets the payment method as per the user's choice.

    Parameters:

    `paymentMethod` - string type
  - **getOrder**

    `public java.util.ArrayList<Item> getOrder()`

This method gets the list of the ordered items as a list.

Returns:

ArrayList of items

- **setOrder**

  `public void setOrder(java.util.ArrayList<Item> order)`

  This method sets the list of the ordered items as a list.

  Parameters:

  `order` - ArrayList of Items

- **addToCart**

  `public void addToCart()`

  This method adds an item to the cart list.

- **displayOrder**

  `public void displayOrder()`

  This method displays order cart items and the total value for the cart.

- **checkout**

  `public void checkout()`

  This method makes the cart empty.

- **choosePayment**

  `public void choosePayment()`

  This method lists out the various payment methods and lets the user select one and then creates an Invoice ID.