[1. 类图展示（详情见附件） 2](#_Toc26837)

[2. 类的属性和方法说明（内部实现） 2](#_Toc32157)

[2.1 Controller（包含购物车、所有图书、所有策略项目；传递参数， 相当于全局变量） 2](#_Toc17351)

[2.2 StringUtil（静态工具类） 3](#_Toc30261)

[2.3 BookSpecification（单本书籍基础信息） 3](#_Toc22748)

[2.4 BookCatalog（图书目录，包含所有图书信息） 3](#_Toc26885)

[2.5 IPricingStrategy（策略接口） 4](#_Toc24161)

[2.6 FlatRateStrategy（绝对值策略） 4](#_Toc25145)

[2.7 PercentageStrategy（百分比策略） 4](#_Toc27291)

[2.8 NoDiscountStrategy（无优惠策略） 5](#_Toc5796)

[2.9 CopositeStrategy（组合策略） 5](#_Toc48)

[2.10 StrategyCatalog（所有策略集合） 5](#_Toc774)

[2.11 PricingStrategyFactory（策略工厂，产生策略） 6](#_Toc25285)

[2.12 Subject（观察主题对象接口） 6](#_Toc22193)

[2.13 Observer（观察者对象接口） 6](#_Toc7586)

[2.14 SaleLineItem（购物车单项商品） 6](#_Toc3105)

[2.15 Sale（购物车所有商品） 7](#_Toc13401)

[2.16 CompositeBestForCustomer（组合策略中获取最优惠的折扣） 7](#_Toc26499)

[3. 类的属性和方法说明（外部UI实现） 7](#_Toc30147)

[3.1 MainUi（主窗口，调出录入书籍、策略维护、购书界面） 7](#_Toc4669)

[3.2 AddBookUi（添加书籍窗口） 7](#_Toc27602)

[3.3 BuyUi（购买窗口，展示所有书籍信息和订单信息） 8](#_Toc1015)

[3.4 ShoppingCartUi（订单信息） 8](#_Toc30475)

[3.5 StrategyUi（策略管理窗口） 8](#_Toc11449)

[3.6 AddStrategy1Ui（百分比策略添加窗口） 8](#_Toc23199)

[3.7 AddStrategy2Ui（绝对值策略添加窗口） 8](#_Toc12562)

[3.8 AddStrategy3Ui（组合策略添加窗口） 8](#_Toc26234)

[3.9 RemoveStrategyUi（删除策略窗口） 9](#_Toc20585)

[3.10 updateStrategy1Ui（百分比策略更新窗口） 9](#_Toc5842)

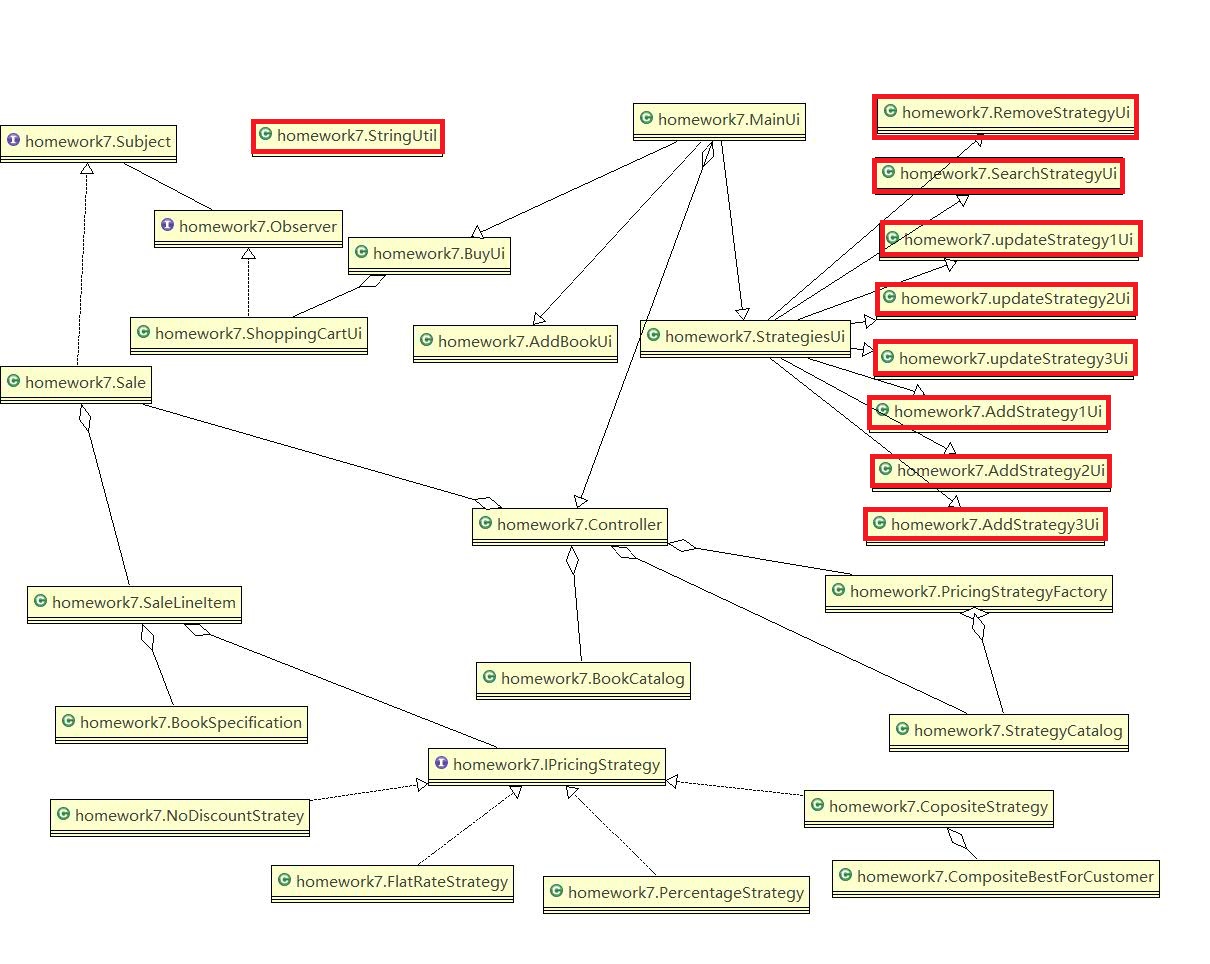
[3.11 updateStrategy2Ui（绝对值策略更新窗口） 9](#_Toc3815)

[3.12 updateStrategy3Ui（组合策略更新窗口） 9](#_Toc31976)

[3.13 SearchStrategyUi（查询策略窗口） 10](#_Toc18329)

# 类图展示（详情见附件）

**注意：红框标识为新加的类**



# 类的属性和方法说明（内部实现）

## Controller（包含购物车、所有图书、所有策略项目；传递参数， 相当于全局变量）

|  |  |
| --- | --- |
| 类名：Controller | |
| 属性名：属性类型 | 属性说明 |
| strateCatalog: StrategyCatalog | 所有策略项目 |
| bookCatalog: BookCatalog | 所有图书项目 |
| sale: Sale | 购物车中所有商品项目 |
| aShoppingCar：Ui\_ShoppingCar | 购物车UI，观察者 |
| 方法名：返回类型 | 方法功能说明 |
| addCompositeStrategy(id: String, name: String, kind: String, bookType: int, strategies: ArrayList<IPricingStrategy>): void | 传入参数，添加新的组合策略 |
| addSimpleStrategy(id: String, name: String, kind: String, bookType: int, profit: double): void | 传入参数，添加新的绝对值优惠策略 |
| addSimpleStrategy(id: String, name: String, kind: String, bookType: int, profit: int): void | 传入参数，添加新的百分比优惠策略 |
| deleteStrategy(mId: String): void | 删除指定策略编号的策略 |
| updateStrategy(id: String, strategy: IPricingStrategy): void | 更新指定策略编号的策略 |
| buyBook(isbn: String, number: int): void | 传入参数，购买书籍 |
| getSale(): Sale | 获取购物车 |
| addBook(bookIsbn: String, bookName: String, bookPrice: double, bookType: int): void | 添加书籍 |

## StringUtil（静态工具类）

|  |  |
| --- | --- |
| 类名：StringUtil（静态工具类） | |
| 方法名：返回类型 | 方法功能说明 |
| isStr(str: String): boolean | 判断字符是否非空 |
| isInteger(str: String): boolean | 判断字符串是否为整数 |
| isDouble(str: String): boolean | 判断字符串s是否为浮点数 |
| isNumber(value: String): boolean | 判断字符串s是否为数字 |

## BookSpecification（单本书籍基础信息）

|  |  |
| --- | --- |
| 类名：BookSpecification | |
| 属性名：属性类型 | 属性说明 |
| isbn: String | 图书ISBN号 |
| price: double | 图书单价 |
| title: String | 图书名称 |
| type: int | 图书类型 |
| 方法名：返回类型 | 方法功能说明 |
| 图书基础信息的get，set函数 | 获取，写入图书基础信息 |
| BookSpecification(isbn: String, name: String, price: double, type: int) | 构造函数，构造相关书籍 |

## BookCatalog（图书目录，包含所有图书信息）

|  |  |
| --- | --- |
| 类名：Book\_Catalog | |
| 属性名：属性类型 | 属性说明 |
| allBooks: ArrayList<BookSpecification> | 所有书籍信息 |
| 方法名：返回类型 | 方法功能说明 |
| getBookSpecification(isbn: String): BookSpecification | 根据指定isbn返回书籍 |

## IPricingStrategy（策略接口）

|  |  |
| --- | --- |
| 类名：IPricingStrategy | |
| 方法名：返回类型 | 方法功能说明 |
| getSubTotal(item: SaleLineItem): double | 获取购物商品item在策略下优惠的价格 |
| getName(): String | 获取该策略的名字 |
| getType(): String | 获取该策略类型 |
| bookTypeToString(): String | 获取该策略适合图书类型 |
| profitToString(): String | 获取该策略的折扣力度 |
| getId(): String | 获取该策略的策略编号 |

## FlatRateStrategy（绝对值策略）

|  |  |
| --- | --- |
| 类名：FlatRateStrategy | |
| 属性名：属性类型 | 属性说明 |
| dicountPerBook：int | 每本优惠额度 |
| name: String | 该策略名称 |
| bookType: int | 该策略适用图书类型 |
| type: String | 该策略类型名称 |
| id: String | 该策略编号 |
| 方法名：返回类型 | 方法功能说明 |
| getSubTotal(item: SaleLineItem): double | 获取购物商品item在该策略下优惠的价格 |
| getName(): String | 获取该策略的名字 |
| getType(): String | 获取该策略类型 |
| bookTypeToString(): String | 获取该策略适合图书类型 |
| profitToString(): String | 获取该策略的折扣力度 |
| getId(): String | 获取该策略的策略编号 |
| 策略基础信息的get，set函数 | 获取，写入图书基础信息 |

## PercentageStrategy（百分比策略）

|  |  |
| --- | --- |
| 类名：PercentageStrategy | |
| 属性名：属性类型 | 属性说明 |
| discountPercentage: int | 百分比优惠 |
| name: String | 该策略名称 |
| bookType: int | 该策略适用图书类型 |
| type: String | 该策略类型名称 |
| id: String | 该策略编号 |
| 方法名：返回类型 | 方法功能说明 |
| getSubTotal(item: SaleLineItem): double | 获取购物商品item在该策略下优惠的价格 |
| getName(): String | 获取该策略的名字 |
| getType(): String | 获取该策略类型 |
| bookTypeToString(): String | 获取该策略适合图书类型 |
| profitToString(): String | 获取该策略的折扣力度 |
| getId(): String | 获取该策略的策略编号 |
| 策略基础信息的get，set函数 | 获取，写入图书基础信息 |

## NoDiscountStrategy（无优惠策略）

|  |  |
| --- | --- |
| 类名：NoDiscountStrategy | |
| 方法名：返回类型 | 方法功能说明 |
| getSubTotal(item: SaleLineItem): double | 获取购物商品item在该策略下优惠的价格 |
| getName(): String | 获取该策略的名字 |
| getType(): String | 获取该策略类型 |
| bookTypeToString(): String | 获取该策略适合图书类型 |
| profitToString(): String | 获取该策略的折扣力度 |
| getId(): String | 获取该策略的策略编号 |

## CopositeStrategy（组合策略）

|  |  |
| --- | --- |
| 类名：CopositeStrategy | |
| 属性名：属性类型 | 属性说明 |
| strategies: ArrayList<IPricingStrategy> | 构成组合策略的简单策略 |
| name: String | 该策略名称 |
| bookType: int | 该策略适用图书类型 |
| type: String | 该策略类型名称 |
| id: String | 该策略编号 |
| cb: CompositeBestForCustomer | 用来获取最优惠策略 |
| 方法名：返回类型 | 方法功能说明 |
| getSubTotal(item: SaleLineItem): double | 获取购物商品item在该策略下优惠的价格 |
| getName(): String | 获取该策略的名字 |
| getType(): String | 获取该策略类型 |
| bookTypeToString(): String | 获取该策略适合图书类型 |
| profitToString(): String | 获取该策略的折扣力度 |
| getId(): String | 获取该策略的策略编号 |
| 策略基础信息的get，set函数 | 获取，写入图书基础信息 |

## StrategyCatalog（所有策略集合）

|  |  |
| --- | --- |
| 类名：StrategyCatalog | |
| 属性名：属性类型 | 属性说明 |
| allStrategies: HashMap<String,IPricingStrategy> | 存储所有策略的map数组 |
| 方法名：返回类型 | 方法功能说明 |

## PricingStrategyFactory（策略工厂，产生策略）

|  |  |
| --- | --- |
| 类名：PricingStrategyFactory | |
| 属性名：属性类型 | 属性说明 |
| *Instance：*PricingStrategyFactory | 单子模式，静态自身 |
| catalog：StrategyCatalog | 所有策略集合，供查重使用 |
| 方法名 | 方法功能说明 |
| getInstance(): PricingStrategyFactory | 单子模式静态调用，实例化自身 |
| getPricingStreategy(bookType: int): IPricingStrategy | 根据输入图书类型生成并返回相对应策略，优先返回组合策略 |
| setCatalog(mcatalog: StrategyCatalog): void | 所有策略传入，不产生新的实例对象 |

## Subject（观察主题对象接口）

|  |  |
| --- | --- |
| 类名：Subject | |
| 属性名：属性类型 | 属性说明 |
| observers: ArrayList<Observer>rver> | e用来存储该主题对象的所有观察者对象 |
| 方法名：返回类型 | 方法功能说明 |
| removeObserver(obser: Observer): void | 移除观察者 |
| registerObserver(obser: Observer): void | 添加观察者 |
| notifyObservers(): void | 通知观察者，更新数据和显示 |

## Observer（观察者对象接口）

|  |  |
| --- | --- |
| 类名：Observer | |
| 属性名：属性类型 | 属性说明 |
| 方法名：返回类型 | 方法功能说明 |
| update(sale: Sale): void | 更新观察者显示，即更新购物车UI显示具体实现 |

## SaleLineItem（购物车单项商品）

|  |  |
| --- | --- |
| 类名：SaleLineItem | |
| 属性名：属性类型 | 属性说明 |
| copies：int | 该类商品购买数量 |
| prodSpec：Book\_Specification | 具体单本书籍基本信息类 |
| strategy：IPricingStrategy | 该本书籍对应的具体策略 |
| 方法名 | 方法功能说明 |
| getSubTotal(): double | 根据策略和书籍定价计算返回优惠价格 |
| SaleLineItem(copy: int, prodspec: BookSpecification) | 构造函数 |
| getProdSpec(): BookSpecification | 获取该订单的书籍 |
| getCopies(): double | 获取该订单的的书籍数量 |

## Sale（购物车所有商品）

|  |  |
| --- | --- |
| 类名：Sale | |
| 属性名：属性类型 | 属性说明 |
| items：ArrayList<SaleLineItem> | 购物车所有单项商品的集合 |
| 方法名 | 方法功能说明 |
| getTotal(): double | 获取购物车所有商品的优惠后的总价 |
| additem(item1: SaleLineItem): void | 添加订单 |
| removeObserver(obser: Observer): void | 移除观察者 |
| registerObserver(obser: Observer): void | 添加观察者 |
| notifyObservers(): void | 通知观察者，更新数据和显示 |

## CompositeBestForCustomer（组合策略中获取最优惠的折扣）

|  |  |
| --- | --- |
| 类名：CompositeBestForCustomer | |
| 属性名：属性类型 | 属性说明 |
| 方法名：返回类型 | 方法功能说明 |
| getSubTotal(item: SaleLineItem, strategies: ArrayList<IPricingStrategy>): double | 获取该订单最优惠的减价 |

# 类的属性和方法说明（外部UI实现）

## MainUi（主窗口，调出录入书籍、策略维护、购书界面）

|  |  |
| --- | --- |
| 类名：MainUi | |
| 方法名：返回类型 | 方法功能说明 |
| onBuyBtn(): void | 购书按钮按下：调出购书窗口 |
| onStrategyBtn(): void | 策略维护按钮按下，调出策略项目窗口 |
| onAddBtn(): void | 添加书籍按钮按下，调出添加书籍窗口 |

## AddBookUi（添加书籍窗口）

|  |  |
| --- | --- |
| 类名：AddBookUi | |
| 方法名：返回类型 | 方法功能说明 |
| onSubmitPerform(): void | 点击确认添加按钮，d调用controller添加书籍方法 |
| onResetPerform(): void | 重置按钮按下，重置输入窗口 |

## BuyUi（购买窗口，展示所有书籍信息和订单信息）

|  |  |
| --- | --- |
| 类名：BuyUi | |
| 方法名：返回类型 | 方法功能说明 |
| onChoseBook(e: MouseEvent): void | 选中书籍，点击加入购物车按钮，将选择书籍信息显示 |
| onBuyBtn(): void | 根据选择书籍及填入书名确认购买 |

## ShoppingCartUi（订单信息）

|  |  |
| --- | --- |
| 类名：ShoppingCartUi | |
| 方法名：返回类型 | 方法功能说明 |
| initTable(): void | 初始化订单表格的信息 |
| update(sale: Sale): void | 根据Sale的改变更新购物车订单 |

## StrategyUi（策略管理窗口）

|  |  |
| --- | --- |
| 类名：StrategyUi | |
| 方法名：返回类型 | 方法功能说明 |
| changeUi(arg0: TreeSelectionEvent): void | 根据java tree不同叶子节点的点击打开不同的窗口 |

## AddStrategy1Ui（百分比策略添加窗口）

|  |  |
| --- | --- |
| 类名：AddStrategy1Ui | |
| 方法名：返回类型 | 方法功能说明 |
| onAddBtn(): void | 写入要添加策略的相应信息，点击确认添加按钮，调用controller简单策略添加的函数 |

## AddStrategy2Ui（绝对值策略添加窗口）

|  |  |
| --- | --- |
| 类名：AddStrategy2Ui | |
| 方法名：返回类型 | 方法功能说明 |
| onAddBtn(): void | 写入要添加策略的相应信息，点击确认添加按钮，调用controller简单策略添加的函数 |

## AddStrategy3Ui（组合策略添加窗口）

|  |  |
| --- | --- |
| 类名：AddStrategy3Ui | |
| 方法名：返回类型 | 方法功能说明 |
| initTable1(): void | 初始化展现所有简单策略的表格 |
| initTable2(): void | 初始化展现已选择所有简单策略的表格 |
| onAddStrategy(): void | 添加在所有简单策略表格中选择的简单策略 |
| onDeleteStrategy(): void | 删除在已选择简单策略表格中选择的简单策略 |
| onSubmitBtn(): void | 根据要添加策略的相应信息，点击确认添加按钮，调用controller组合策略添加的函数 |

## RemoveStrategyUi（删除策略窗口）

|  |  |
| --- | --- |
| 类名：RemoveStrategyUi | |
| 方法名：返回类型 | 方法功能说明 |
| selectItem(arg0: MouseEvent): void | 根据所有策略表格中选择的策略更新信息 |
| onRemoveBtn(): void | 删除鼠标点击的策略 |
| initTable(): void | 初始化表格，展示所有策略 |

## updateStrategy1Ui（百分比策略更新窗口）

|  |  |
| --- | --- |
| 类名：updateStrategy1Ui | |
| 方法名：返回类型 | 方法功能说明 |
| initTable(): void | 初始化表格，展示所有百分比策略 |
| onChoseBook(arg0: MouseEvent): void | 根据所有百分比策略表格中选择的策略更新信息 |
| onSubmitBtn(): void | 提交更新策略，调用controller的百分比策略更新函数 |

## updateStrategy2Ui（绝对值策略更新窗口）

|  |  |
| --- | --- |
| 类名：updateStrategy2Ui | |
| 方法名：返回类型 | 方法功能说明 |
| initTable(): void | 初始化表格，展示所有绝对值策略 |
| onChoseBook(arg0: MouseEvent): void | 根据所有绝对值策略表格中选择的策略更新信息 |
| onSubmitBtn(): void | 提交更新策略，调用controller的绝对值策略更新函数 |

## updateStrategy3Ui（组合策略更新窗口）

|  |  |
| --- | --- |
| 类名：updateStrategy3Ui | |
| 方法名：返回类型 | 方法功能说明 |
| initTable1(): void | 初始化表格1，展示所有组合策略 |
| initTable2(): void | 初始化表格2，展示所有简单策略 |
| initTable3(): void | 初始化表格3，展示所有选中的组合策略所包含的简单策略 |
| onChoseBook(arg0: MouseEvent): void | 根据所有组合策略表格中选择的策略更新信息 |
| onAddStrategy(): void | 添加在所有简单策略表格中选择的简单策略 |
| onDeleteStrategy(): void | 删除在已选择简单策略表格中选择的简单策略 |
| onSubmitBtn(): void | 提交更新策略，调用controller的组合策略更新函数 |

## SearchStrategyUi（查询策略窗口）

|  |  |
| --- | --- |
| 类名：Ui\_ShoppingCar | |
| 方法名 | 方法功能说明 |
| initTable(): void | 初始化表格，展示所有策略 |
| onChose(arg0: MouseEvent): void | 根据所有组合策略表格中选择的策略更新信息 |
| onSubmitBtn(): void | 根据输入框内容，鼠标锁定表格中对应的策略 |