北京邮电大学 计算机学院

《Java语言程序设计》实验报告

姓名 王睿嘉

学号2015211906

班级2015211307

实验 设计模式

1. 实验内容和环境描述
2. 实验内容

设计一个贩卖各类书籍的电子商务系统。

1. 环境描述

编程语言：Java

集成开发环境：IntelliJ IDEA

SDK版本：1.8

1. 实验设计
2. 整体说明

类型层次：[Type Hierarchy.svg](Type%20Hierarchy.svg)

具体类图：<Details.svg>

1. 关键点说明

|  |  |
| --- | --- |
| public class AddBookUI extends JFrame implements ActionListener | |
| 属性名 | 属性说明 |
| Controller controller | 控制器，通过它与其余模块进行交互 |
| 方法名 | 功能说明 |
| public void actionPerformed(ActionEvent) | 响应各种事件 |

|  |  |
| --- | --- |
| public class BookCatalog | |
| 属性名 | 属性说明 |
| ArrayList<BookSpecification> books | 所有图书信息 |
| 方法名 | 功能说明 |
| public void addBook(BookSpecification) | 添加图书 |
| public ArrayList<BookSpecification> getBooks() | 获得所有图书信息 |
| public BookSpecification getBookSpecification(String) | 通过ISBN号码，获得相应图书信息 |

|  |  |
| --- | --- |
| public class BookSpecification | |
| 属性名 | 属性说明 |
| String isbn | ISBN号码 |
| String title | 标题 |
| double price | 价格 |
| BookType type | 类型 |
| 方法名 | 功能说明 |
| public String getIsbn() | 获得ISBN号码 |
| public String getTitle() | 获得标题 |
| public double getPrice() | 获得价格 |
| public BookType getType() | 获得类型 |

|  |  |
| --- | --- |
| public enum BookType | |
| NON\_TEXTCOMPUTER | 非教材计算机类 |
| TEXTBOOK | 教材类 |
| COMICS | 连环画类 |
| HEALTHCARE | 养生类 |
| OTHERS | 其他类 |

|  |  |
| --- | --- |
| public class BuyUI extends JFrame implements ActionListener | |
| 属性名 | 属性说明 |
| Controller controller | 控制器，通过它与其余模块进行交互 |
| BookSpecification bookSpecification | 所购买图书的信息 |
| 方法名 | 功能说明 |
| public void actionPerformed(ActionEvent e) | 响应各种事件 |

|  |  |
| --- | --- |
| public class CompositeBestForCustomer extends CompositeStrategy | |
| 方法名 | 功能说明 |
| public double getSubTotal(SaleLineItem) | 计算最优价格 |

|  |  |
| --- | --- |
| public abstract class CompositeStrategy implements IPricingStrategy | |
| 属性名 | 属性说明 |
| ArrayList<IPricingStrategy> strategies | 所含的简单策略 |
| StrategyType strategyType | 策略类型 |
| int strategyNumber | 策略编号 |
| 方法名 | 功能说明 |
| public abstract double getSubTotal(SaleLineItem) | 计算最优价格 |
| public StrategyType getStrategyType() | 获得策略类型 |
| public int getStrategyNumber() | 获得策略编号 |
| public void add(IPricingStrategy) | 增添简单策略 |

|  |  |
| --- | --- |
| public class Controller implements SubjectForMain | |
| 属性名 | 属性说明 |
| BookCatalog bookCatalog | 图书目录 |
| StrategyCatalog strategyCatalog | 策略目录 |
| Sale sale | 销售记录 |
| PricingStrategyFactory pricingStrategyFactory | 策略工厂 |
| ArrayList<ObserverForMain> observerForMains | 所有观察者 |
| 方法名 | 功能说明 |
| public void removeObserver(ObserverForMain) | 移除观察者 |
| public void registerObserver(ObserverForMain) | 注册观察者 |
| public void notifyObservers() | 通知观察者 |
| public void addBook(BookSpecification) | 添加图书 |
| public boolean addCompositeStrategy(BookType ,  int,String) | 添加组合策略 |
| public boolean addSimpleStrategy(BookType,  StrategyType,int,String) | 添加简单策略 |
| public void deleteStrategy(int) | 删除策略 |
| public boolean updateStrategy(BookType,  StrategyType,int,String) | 更改策略 |
| public boolean buyBook(int copies,String isbn) | 购买图书 |
| public Sale getSale() | 获得销售记录 |
| public BookSpecification getBook(String) | 通过ISBN号码，获得相应图书信息 |
| public IPricingStrategy returnStrategies(int) | 通过编号，获得相应策略 |
| public BookCatalog getBookCatalog() | 获得图书目录 |
| public StrategyCatalog getStrategyCatalog() | 获得策略目录 |

|  |  |
| --- | --- |
| public class FlatRateStrategy implements IPricingStrategy | |
| 属性名 | 属性说明 |
| int discountPerBook | 每本书的减免价格 |
| StrategyType strategyType | 策略类型 |
| int strategyNumber | 策略编号 |
| 方法名 | 功能说明 |
| public double getSubTotal(SaleLineItem) | 计算优惠后总价 |
| public StrategyType getStrategyType() | 获得策略类型 |
| public void setStrategyNumber(int) | 设置策略编号 |
| public void setDiscountPerBook(int) | 设置每本书的减免价格 |
| public int getStrategyNumber() | 获得策略编号 |

|  |  |
| --- | --- |
| public interface IPricingStrategy | |
| 方法名 | 功能说明 |
| double getSubTotal(SaleLineItem) | 计算优惠后总价 |
| StrategyType getStrategyType() | 获得策略类型 |
| int getStrategyNumber() | 获得策略编号 |

|  |  |
| --- | --- |
| public class MainUI extends JFrame implements ObserverForMain,ActionListener | |
| 属性名 | 属性说明 |
| Controller controller | 控制器，通过它与其余模块进行交互 |
| ShoppingCartUI shoppingCartUI | 购物车界面 |
| class TableModel extends DefaultTableModel | 内部类，以表格形式显示图书目录 |
| 方法名 | 功能说明 |
| public void update() | 观察者自更新 |
| public void actionPerformed(ActionEvent) | 响应各种事件 |
| public void init() | 初始化样例 |

|  |  |
| --- | --- |
| public class NoDiscountStrategy implements IPricingStrategy | |
| 属性名 | 属性说明 |
| StrategyType strategyType | 策略类型 |
| 方法名 | 功能说明 |
| public double getSubTotal(SaleLineItem) | 计算优惠后总价 |
| public StrategyType getStrategyType() | 获得策略类型 |
| public int getStrategyNumber() | 获得策略编号 |

|  |  |
| --- | --- |
| public interface Observer | |
| 方法名 | 功能说明 |
| void update(Sale) | 观察者自更新 |

|  |  |
| --- | --- |
| public interface ObserverForMain | |
| 方法名 | 功能说明 |
| void update() | 观察者自更新 |

|  |  |
| --- | --- |
| public class PercentageStrategy implements IPricingStrategy | |
| 属性名 | 属性说明 |
| double discountPercentage | 百分比折扣 |
| StrategyType strategyType | 策略类型 |
| int strategyNumber | 策略编号 |
| 方法名 | 功能说明 |
| public double getSubTotal(SaleLineItem) | 计算优惠后总价 |
| public StrategyType getStrategyType() | 获得策略类型 |
| public int getStrategyNumber() | 获得策略编号 |
| public void setStrategyNumber(int) | 设置策略编号 |
| public void setDiscountPercentage(double) | 设置百分比折扣 |

|  |  |
| --- | --- |
| public class PricingStrategyFactory | |
| 属性名 | 属性说明 |
| PricingStrategyFactory instance | 实例 |
| StrategyCatalog catalog | 策略目录 |
| 方法名 | 功能说明 |
| public PricingStrategyFactory getInstance() | 获得实例 |
| public IPricingStrategy getPricingStrategy(BookType) | 通过图书类型，获得相应策略信息 |
| public void setCatalog(StrategyCatalog) | 设置策略目录 |

|  |  |
| --- | --- |
| public class Sale implements Subject | |
| 属性名 | 属性说明 |
| ArrayList<SaleLineItem> items | 销售记录条项 |
| ArrayList<Observer> observers | 所有观察者 |
| 方法名 | 功能说明 |
| public void removeObserver(Observer) | 移除观察者 |
| public void registerObserver(Observer) | 注册观察者 |
| public void notifyOberservers() | 通知观察者 |
| public double getTotal() | 计算优惠后总价 |
| public void addSaleLineItem(SaleLineItem) | 增添销售记录 |
| public ArrayList<SaleLineItem> getItems() | 获得销售记录条项 |

|  |  |
| --- | --- |
| public class SaleLineItem | |
| 属性名 | 属性说明 |
| int copies | 购买数量 |
| BookSpecification prodSpec | 图书信息 |
| IPricingStrategy strategy | 策略信息 |
| 方法名 | 功能说明 |
| public double getSubTotal() | 计算优惠后总价 |
| public BookSpecification getProdSpec() | 获得图书信息 |
| public int getCopies() | 获得购买数量 |
| public IPricingStrategy getStrategy() | 获得策略信息 |

|  |  |
| --- | --- |
| public class ShoppingCartUI extends JFrame implements Observer,ActionListener | |
| 属性名 | 属性说明 |
| Subject subject | 被观察者 |
| 方法名 | 功能说明 |
| public void update(Sale) | 观察者自更新 |
| public void actionPerformed(ActionEvent) | 响应各种事件 |
| class TableModel extends DefaultTableModel | 内部类，以表格形式显示销售记录 |

|  |  |
| --- | --- |
| public class StrategiesUI extends JFrame implements ActionListener | |
| 属性名 | 属性说明 |
| Controller controller | 控制器 |
| StrategyType strategyType | 策略类型 |
| BookType bookType | 图书类型 |
| 方法名 | 功能说明 |
| public void actionPerformed(ActionEvent) | 响应各种事件 |
| class TableModel extends DefaultTableModel | 内部类，以表格形式显示策略目录 |

|  |  |
| --- | --- |
| public class StrategyCatalog | |
| 属性名 | 属性说明 |
| HashMap<BookType,IPricingStrategy> strategies | 与图书类型相关联的策略信息 |
| 方法名 | 功能说明 |
| HashMap<BookType,IPricingStrategy> getStrategies() | 获得与图书类型相关联的策略信息 |
| public BookType returnBookType(int) | 通过策略编号，获得相应图书类型 |
| public StrategyType returnStrategyType(int) | 通过策略编号，获得相应策略类型 |
| boolean hasStrategy(int) | 判断策略编号是否存在 |
| public IPricingStrategy returnStrategies(int) | 通过策略编号，获得相应策略信息 |

|  |  |
| --- | --- |
| public enum StrategyType | |
| FLATRATE | 绝对值优惠 |
| PERCENTAGE | 百分比折扣优惠 |
| COMPOSITE | 组合优惠 |
| NO | 无优惠 |

|  |  |
| --- | --- |
| public interface Subject | |
| 方法名 | 功能说明 |
| void removeObserver(Observer) | 移除观察者 |
| void registerObserver(Observer) | 注册观察者 |
| void notifyOberservers() | 通知观察者 |

|  |  |
| --- | --- |
| public interface SubjectForMain | |
| 方法名 | 功能说明 |
| void removeObserver(ObserverForMain) | 移除观察者 |
| void registerObserver(ObserverForMain) | 注册观察者 |
| void notifyOberservers() | 通知观察者 |

|  |  |
| --- | --- |
| public class Tester | |
| 方法名 | 功能说明 |
| public static void main(String[]) | 程序入口 |

1. 实验心得

本实验既包含了观察者、工厂、策略、组合、单子等设计模式，又包含了图形化、类继承与接口等具

体内容，综合了本学期所教授的Java语言知识。

在编程过程中，主要心得有以下五点：

* 1. 题目。对照实验指导文档所含的类图逐一建立.java文件，并根据模块交互方式理解程序流程。起初，极为艰难。反复阅读，反复体会，才逐渐了解框架；
  2. 设计模式。由于课时原因，设计模式是课堂并未涉及的模块。通过自学，了解观察者、工厂等模式的内涵，惊叹于Java的强大与实用；
  3. 图形化。Java自身集成了图形化。在实现上，与Qt有许多共通点，但也存在差异。较为熟悉Qt布局及美化方式，而在本次实验中并不完全适用，加之时间的局限性，界面较为简陋；
  4. 数据结构。HashMap、ArrayList等接近于Python数据结构，减少了代码量，提高了运行效率；
  5. 错误处理。此次实验中存在较多差错场景，如信息不完整、格式不正确、表格操作不当等，需要一一考虑。Baidu过程中，发现consume函数极为好用，可以直接规范输入内容，省去了差错情形的大量判断。同时，善用try…catch捕获异常，又为错误处理带来极大便利；

实践出真知，本次实验是对课堂和书本所学知识的补充。实际情形与已了解到的原理大体一致，但又

复杂许多。通过自己动手、亲力亲为编写代码，加深了对Java语言的理解和记忆，收获颇丰。