Java 课程设计

潮汕火锅自助点菜系统

课程名称：JAVA 与面向对象编程

小组组号：3

小组成员　　学号

郑有为 19335286

曾家洋 19335008

黄皓佳 19309049

（顺序不分先后）

日期：　2021 年 1 月 2 日

目录

1. 项目背景介绍

互联网时代，越来越多的餐厅，饮品店提供自助点餐的系统：我们可以看到海底捞用平板+自助点餐系统的方式替代了传统的纸质点单/服务员点单，KFC，麦当劳，各式饮品店也有提供手机上的线上点餐系统。电子点单显然能够节省人力物力，方便管理，使用电子点单已成为餐饮行业的趋势。

2. 系统功能介绍

本系统为餐厅开发的一套在线点餐管理系统，旨在合理化安排餐厅的工作，提高餐厅的管理效率，同时也提高客户的就餐体验。

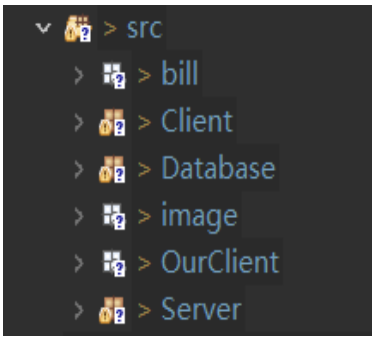
本系统分为客户端和服务端，客户端的功能主要是为客户提供注册登录并实现在线点餐，将点餐后的信息发送至服务端，并等待服务端的响应。而服务端的功能则是管理餐品的库存并且接受来自客户端的订单通知，服务端可根据库存量或其他因素，对客户发来的订单选择接受或拒绝，并将处理结果发送至客户端。

因此，整个系统的运作，实际上与现实中客户点餐与餐厅服务之间的交互是一致的。

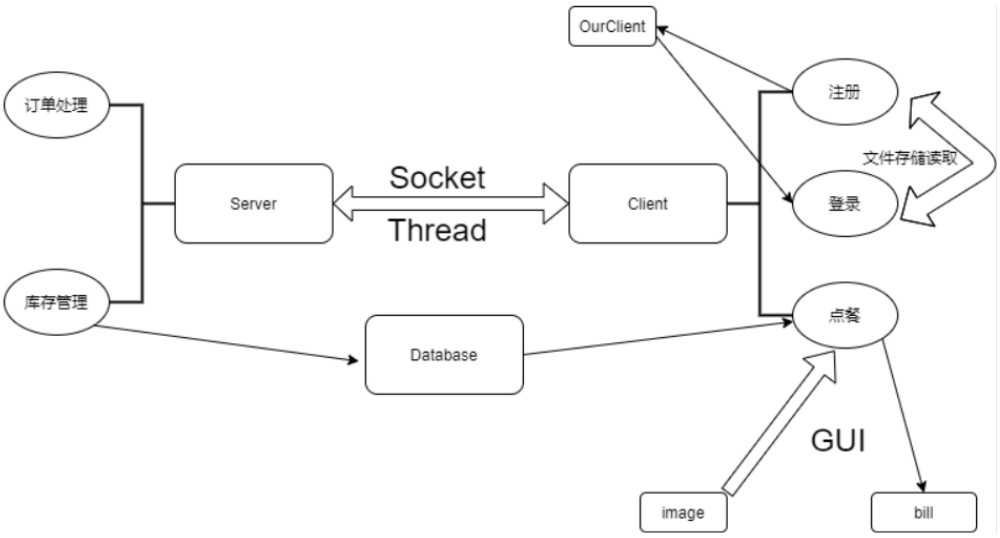
而对于程序的具体使用说明，我们已经在展示 PPT 中呈现，不再赘述。

3. 系统类图

如下图所示，我们程序分了六个部分，bill 存放的是用户点餐后生成的账单，image 存放的是我们所用到的图片，OurClient 存放的是账户信息，Datebase 主要是有关食物的信息，Server 实现了服务端，Client 实现了客户端。



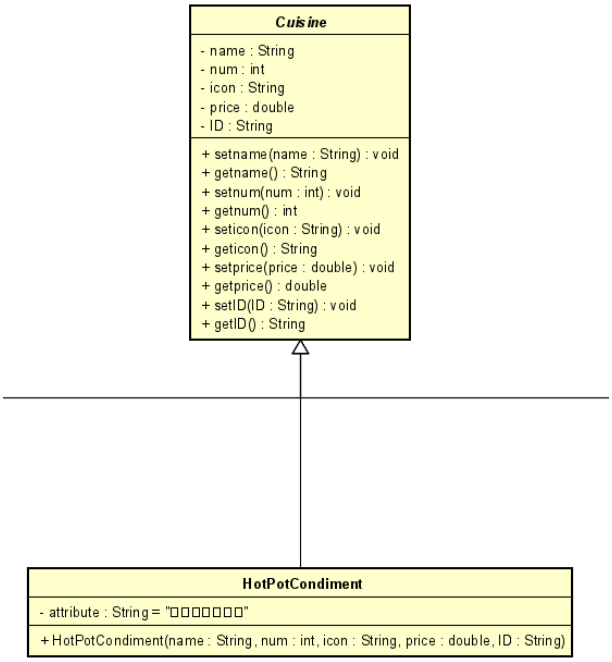
他们之间的功能和联系如下图所示，Server 实现的功能有订单的处理和库存的管理，Client 实现的功能有用户的注册、登录以及点餐，注册和登录通过文件的存储读取实现，文件便是存放在 OurClient 中。Server 和 Client 之间通过网络编程以及线程建立连接。Datebase 则连接了服务端的库存管理以及 Client 的点餐。用户点完餐之后会将点餐信息存放在 bill 中。Image 中的图片则是实现点餐界面。



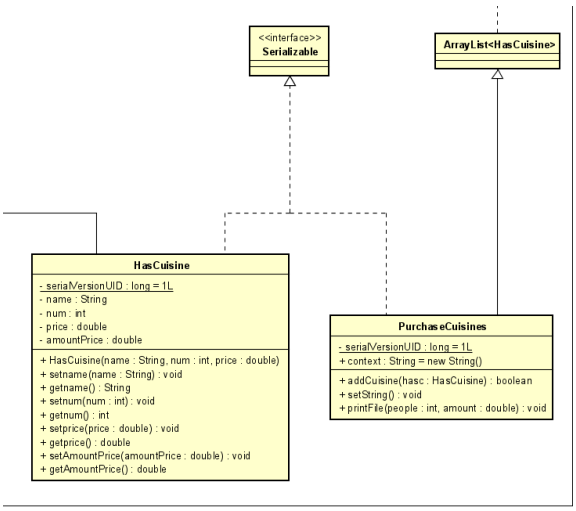
主要模块的 UML 说明：

下面几张 UML 局部图是程序最重要的几个类的 UML，由于程序整体构架不够合理，UML 整体图比较大，所以没有附上整体 UML 图。

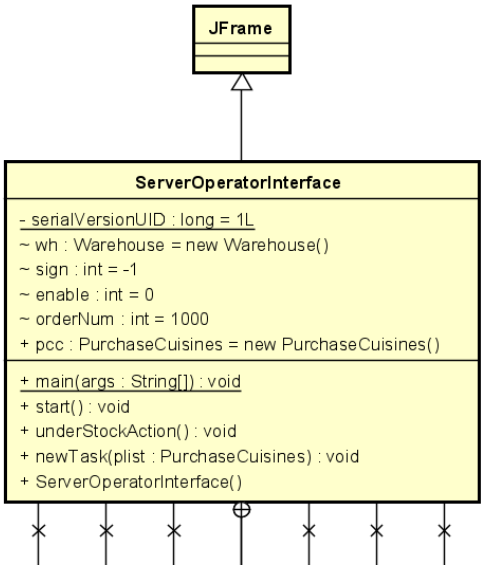
1. 菜品类，其作为几个具体菜品类的基类



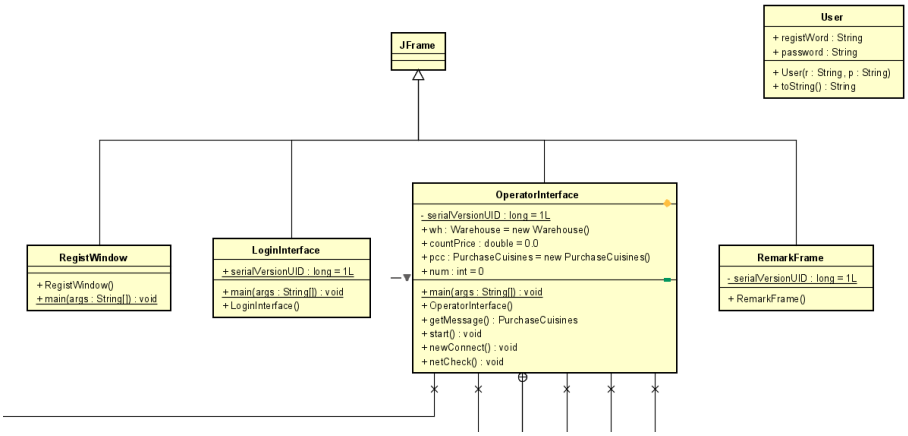
2. 已点菜品类和点菜类，后者继承了已点菜品类的链表，记录一单所点的所有菜品和数量，PurchaseCuisines 类作为数据从客户端传输到服务端



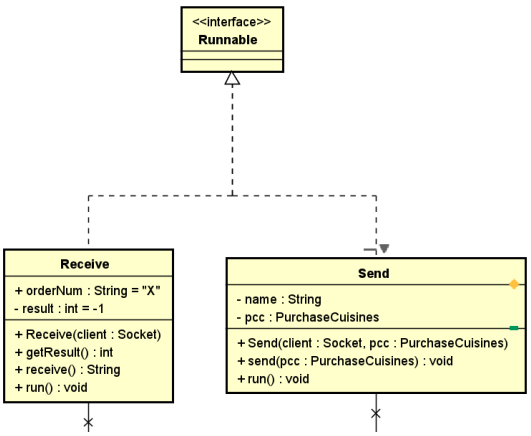
3. 服务端界面类



4. 客户端界面类：主要包括点菜界面类，登陆界面类和用户类等

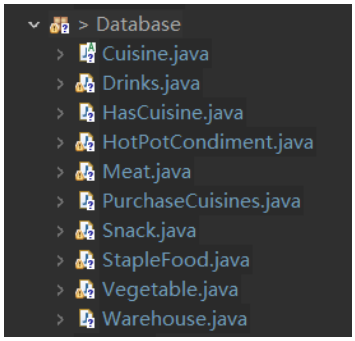


5. 线程与网络编程部分，Send 和 Receive 类

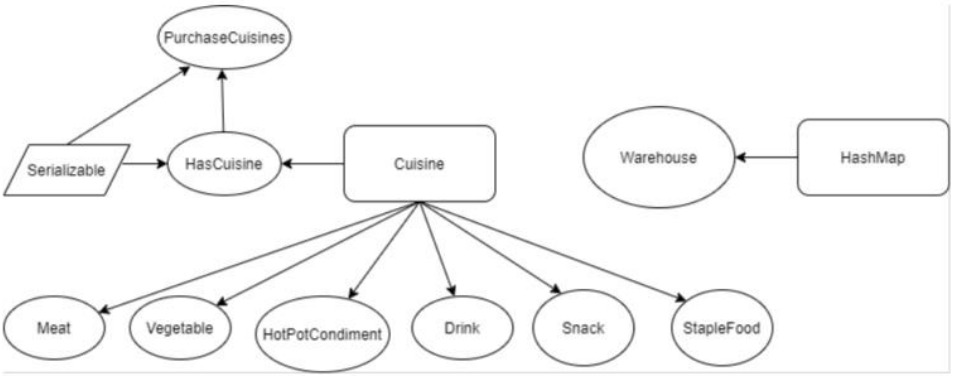


4. 关键模块说明

1. Database 模块：



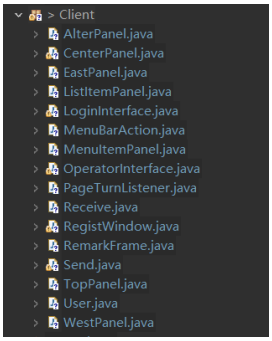
层次结构如下图



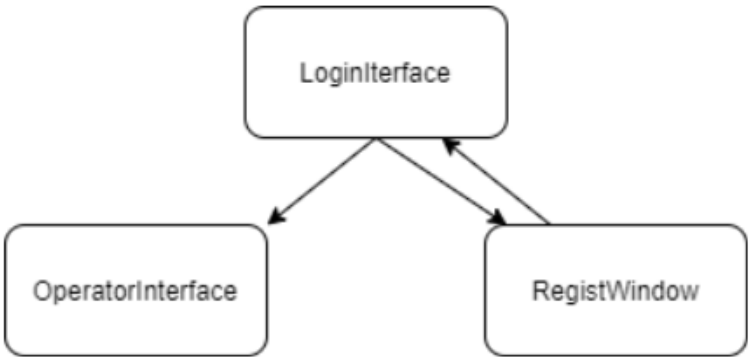
首先有一个 Cuisine 类，是一个抽象类，然后下面这些是各种种类的食物对应的类，就是继承了这个抽象类。然后还有一个 HasCuisne 类，也继承了 Cuisine 类，然后还实现了 Serializable 接口，然后 PurchaseCuisines 类则是继承了一个 Cuisine 类，也实现了 Serializable 接口，它的功能是记录顾客点单的信息。

然后还有一个独立的 Warehouse 类，它利用哈希表实现了菜品仓库。

2. Client 模块：



其中一部分类分别实现了点餐界面的各模块，OperatorInterface 中有主函数，运行之后弹出该界面。然后 LoginInterface 和 RegistWindow 中也各自有主函数，运行之后则是弹出这两个窗口，三者之间的关系如下图，也就是直接登录或者注册后再登录。



最后就是 Send 和 Receive 类。Send 和 Receive 类为向服务端发送及接受信息，均继承 Runnable，故发送和接受为两个线程，也就是说是并行的。

3. Server 模块：

Server 模块的实现与 Client 模块大同小异，也是一些类实现界面布局，另外一些类为一些监听器，实现底层的逻辑功能。

五：知识点应用说明

1.类和对象：

对于 java 来说，基本上所有的都涉及类和对象 。

2.超类和继承：

(1) 超类：

菜品类 Cuisine 作为 Meat，Vegetable 等几个食物类的超类。

(2) 继承：

a. DataBase 所有类中除了 HasCuisines.java 没有继承 Cuisine 外，其他类都有继承这个类（主要用于实现多种多样的菜品）；

b.AlterPanel,CenterPanel,EastPanel,WestPanel,ListItemPanel,MenuItemPanel,TopPanel 都继承了 JPanel；

c.RegisterWindow,LoginInterface,OperatorInterface,RemarkFrame 都继承了 JFrame。

(Client 和 Server 两个 package 中都类似，Server 端不赘述)

3. 接口及其实现：

主要体现在我们实现已有的接口，

MenuBarAction 实现了 FocusListener 的接口；

PageTurnListener 实现了 ActionListener 的接口；

Receive 和 Send 实现了 Runnable 的接口；

HasCuisine，PurchaseCuisine 实现了 Serializable 的接口。

4. 异常处理：

在多个类的方法中 catch 了 IOException，FileNotFoundException，ClassNotFoundException。（主要是在数据的处理上去 try 和 catch exception）

再顾客点单时，如果对于用户用餐人数为零或者点餐内容为空则会当作异常处理，返回 Jdialog 提示顾客·。

5. 多线程：

在 Send 和 Receive 这两个类中实现了 Runnable 这个接口；

线程一是通过开启多个用户来调用那些类和函数来实现，二是体现在 Receive 和 Send 同步运行，实现交互的同步性。

6. 文件存储：

我们分了 bill 和 OurClient 两个文件包存储，bill 存储客户点单信息，OurClient 存储客户信息。

在用户登陆注册时需要对客户数据文件 OurClient 进行读写，再客户请求结账时，我们将用户点单信息写入新建账单文件中并储存在 bill 文件夹里。

7. 网络编程:

我们采用 TCP 协议，将网络编程分为两部分，一是服务端的 Server Socket 部分，我们对应的界面类中是实现了发送与接受的方法，二是用户端的 Socket 部分，我们在 Send 和 Receive 这两个类中分别实现了用户端数据的发送与接收。

8. 拓展:

(1). HashMap: 我们主要是在 WareHouse 类上使用了 HashMap 进行存储。

(2). JavaGUI: 我们在登录，注册，客户端以及服务端全都使用了基于 Swing 的图形化界面，适当地考虑了人性化的界面排版与配色。

六：创新点与技术难点说明

创新点:

1. 使用了 HashMap 来存储每一条菜品，我们以菜品编号为 Key, 简化了菜品遍历过程

2. 我们使用文件的读写来模拟账单的打印，账单以打单时间命名，记录用户的用餐人数，所选菜品和消费总额。

3. 使用了 GUI 界面来优化整个点餐过程和后台管理

技术难点说明:

1.在处理线程和网络编程的结合与协调上下了功夫，我们建立以了一个管理用户连接的链表，使用发送与接收双线程来处理服务端与客户端信息的传送。

2. GUI 的设计和监听器的设计与逻辑相应，不同于简单对话软件或者简单游戏，我们的按钮与显示的逻辑十分复杂，我们采用分块处理的基本方式，并在主类上设置所有的监听器来实现各界面模块的交互。

七： 未解决问题与难点讨论

由于时间的限制，我们尚有很多功能没有实现完全，有些界面回应的逻辑没有完善，下面是部分待完善的功能与逻辑实现：

1. 我们原本设定用户点餐后可以积累积分，存储在用户信息当中，餐厅可以推出积分福利与优惠，尚未实现；

2. 用户点餐备注功能没有实现完全，暂时还不能返回此信息至服务端；

3. 用户信息缺乏安全保障：我们起初设定用户账号信息应该又服务端管理，放在服务端一侧，防止用户窃取信息，但由于时间的缘故没有将其实现。

4. 用户点餐界面的点餐逻辑存在漏洞，由于 setEnable 的机制没有梳理清楚，用户可以再等待后台相应的时候先点击记账或强制修改菜品而触发一系列未定义行为，我们尚未对这些不合理操作设置异常处理。

5. 服务端每次只能接受一单，只有在处理完一单后才能处理下一单，与现实中的接单机制不符合。

6. 用户多次点餐后会出现，后台回应与点餐界面回应不同步的问题，还没有解决，即我们只能保证前几次用户点餐流程的正确性。