

# 华中科技大学

# 课程实验报告

课程名称： Java 语言程序设计

实验名称： 医院简易挂号管理系统

院 系： 计算机科学与技术

专业班级： 信息安全 1503 班

学 号： U201514858

姓 名： 陈 薇

指导教师： 吕新桥

2018 年 05 月 22 日

## 一、需求分析

### 1. 题目要求

采用桌面应用程序模式，开发一个医院挂号系统，管理包括人员、号种及其挂号费用，挂号退号等信息，完成登录、挂号、查询和统计打印功能。数据库表如下所示，建立索引的目的是加速访问，请自行确定每个索引要涉及哪些字段。

**T\_KSXX (科室信息表)**

字段名称	字段类型	主键	索引	可空	备注
KSBH	CHAR(6)	是	是	否	科室编号，数字
KSMC	CHAR(10)	否	否	否	科室名称
PYZS	CHAR(8)	否	否	否	科室名称的拼音字首

**T\_BRXX (病人信息表)**

字段名称	字段类型	主键	索引	可空	备注
BRBH	CHAR(6)	是	是	否	病人编号，数字
BRMC	CHAR(10)	否	否	否	病人名称
DLKL	CHAR(8)	否	否	否	登录口令
YCJE	DECIMAL(10,2)	否	否	否	病人预存金额
DLRQ	DateTime	否	否	是	最后一次登录日期及时间

**T\_KSYS (科室医生表)**

字段名称	字段类型	主键	索引	可空	备注
YSBH	CHAR(6)	是	是	否	医生编号，数字，第 1 索引
KSBH	CHAR(6)	否	是	否	所属科室编号，第 2 索引
YSMC	CHAR(10)	否	否	否	医生名称
PYZS	CHAR(4)	否	否	否	医生名称的拼音字首
DLKL	CHAR(8)	否	否	否	登录口令
SFZJ	BOOL	否	否	否	是否专家
DLRQ	DATETIME	否	否	是	最后一次登录日期及时间

**T\_HZXX (号种信息表)**

字段名称	字段类型	主键	索引	可空	备注
HZBH	CHAR(6)	是	是	否	号种编号，数字，第 1 索引
HZMC	CHAR(12)	否	否	否	号种名称

PYZS	CHAR(4)	否	否	否	号种名称的拼音字首
KSBH	CHAR(6)	否	是	否	号种所属科室，第 2 索引
SFZJ	BOOL	否	否	否	是否专家号
GHRs	INT	否	否	否	每日限定的挂号人数
GHFY	DECIMAL(8,2)	否	否	否	挂号费

T\_GHXX (挂号信息表)

字段名称	字段类型	主键	索引	可空	备注
GHBH	CHAR(6)	是	是	否	挂号的顺序编号，数字
HZBH	CHAR(6)	否	是	否	号种编号
YSBH	CHAR(6)	否	是	否	医生编号
BRBH	CHAR(6)	否	是	否	病人编号
GHRC	INT	否	是	否	该病人该号种的挂号人次
THBZ	BOOL	否	否	否	退号标志=true 为已退号码
GHFY	DECIMAL(8,2)	否	否	否	病人的实际挂号费用
RQSJ	DATETIME	否	否	否	挂号日期时间

为了减少编程工作量，T\_KSXX、T\_BRXX、T\_KSYS、T\_HZXX 的信息手工录入数据库，每个表至少录入 6 条记录，所有类型为 CHAR(6)的字段数据从“000001”开始，连续编码且中间不得空缺。为病人开发的桌面应用程序要实现的主要功能具体如下：

(1) 病人登录：输入自己的病人编号和密码，经验证无误后登录。

(2) 病人挂号：病人处于登录状态，选择科室、号种和医生（非专家医生不得挂专家号，专家医生可以挂普通号）；输入缴费金额，计算并显示找零金额后完成挂号。所得挂号的编号从系统竞争获得生成，挂号的顺序编号连续编码不得空缺。

功能（2）的界面如下所示，在光标停在“科室名称”输入栏时，可在输入栏下方弹出下拉列表框，显示所有科室的“科室编号”、“科室名称”和“拼音字首”，此时可通过鼠标点击或输入科室名称的拼音字首两种输入方式获得“科室编号”，用于插入 T\_GHXX 表。注意，采用拼音字首输入时可同时完成下拉列表框的科室过滤，使得下拉列表框中符合条件的科室越来越少，例如，初始为“内一科”和“内二课”。其它输入栏，如“医生姓名”、“号种类别”、“号种名称”也可同时支持两种方式混合输入。

每种号种挂号限定当日人次，挂号人数超过规定数量不得挂号。一个数据一致的程序要保证：挂号总人数等于当日各号种的挂号人次之和，病人的账务应保证开支平衡。已退号码不得用于重新挂号，每个号重的 GHRC 数据应连续不间断，GHRC 从 1 开始。若病人有预存金额则直接扣除挂号费，此时“交款金额”和“找零金额”处于灰色不可操作状态。

## 门诊挂号

科室名称

医生姓名

号种类别

号种名称

交款金额

应缴金额

找零金额

挂号号码

为医生开发的桌面应用程序要实现的主要功能具体如下：

（1）医生登录：输入自己的医生编号和密码，经验证无误后登录。

（2）病人列表：医生处于登录状态，显示自己的挂号病人列表，按照挂号编号升序排列。

显示结果如下表所示。

挂号编号	病人名称	挂号日期时间	号种类别
000001	章紫衣	2018-12-30 11:52:26	专家号
000003	范冰冰	2018-12-30 11:53:26	普通号
000004	刘德华	2018-12-30 11:54:28	普通号

（3）收入列表：医生处于登录状态，显示所有科室不同医生不同号种起止日期内的收入合计，起始日期不输入时默认为当天零时开始，截止日期至当前时间为止。时间输入和显示结果如下表所示。

起始时间：2018-12-30 00:00:00    截止时间：2018-12-30 12:20:00

科室名称	医生编号	医生名称	号种类别	挂号人次	收入合计
感染科	000001	李时珍	专家号	24	48
感染科	000001	李时珍	普通号	10	10
内一科	000002	扁鹊	普通号	23	23
保健科	000003	华佗	专家号	10	20

病人应用程序和医生应用程序可采用主窗口加菜单的方式实现。例如，医生应用程序有三个菜单项，分别为“病人列表”、“收入列表”和“退出系统”等。

考虑到客户端应用程序要在多台计算机上运行，而这些机器的时间各不相同，客户端程序每次在启动时需要同数据库服务器校准时间，可以建立一个时间服务程序或者直接取数据库时间校准。建议大家使用 **MS SQL** 数据库开发。

挂号时锁定票号可能导致死锁，为了防止死锁或系统响应变慢，建议大家不要锁死数据库表或者字段。程序编写完成后，同时启动两个挂号程序进行单步调试，以便测试两个病人是否会抢到同一个号、或者有号码不连续或丢号的现象。

系统考核目标：（1）挂号后数据库数据包括挂号时间不会出现不一致或时序颠倒现象，以及挂号人次超过该号种当日限定数量的问题；（2）挂号号码和挂号人次不会出现不连续或丢号问题；（3）病人的开支应平衡，并应和医院的收入平衡；（4）系统界面友好、操作简洁，能支持全键盘操作、全鼠标操作或者混合操作；（5）能支持下拉列表框过滤输入；（6）系统响应迅速，不会出现死锁；（7）统计报表应尽可能不采用多重或者多个循环实现；（8）若采用时间服务器程序校准时间，最好能采用心跳检测机制，显示客户端的上线和下线情况。

思考题：当病人晚上 11:59:59 秒取得某号种的挂号价格 10 元，当他确定保存时价格在第 2 天 00:00:00 已被调整为 20 元，在编程时如何保证挂号费用与当天价格相符？

## 2. 需求分析

本次实验需要实现为医生和病人开发的桌面应用程序，完成病人登陆、病人挂号、医生登陆、医生查看病人信息、收入信息等。具体应实现功能如下：

### 1、建立数据库基本表

根据题目要求，建立 T\_KSXX、T\_BRXX、T\_KSYS、T\_HZXX、T\_GHXX 五个基本表，并对其中前四个基本表进行信息的初始录入。这四张表中，每张表至少有 6 条记录，所有 CHAR(6) 类型的字段数据均是从“000001”开始编号并且数据连续中间不得有空缺。

### 2、设计病人系统的用户界面和功能

病人系统的用户界面可以划分为三个：病人登陆、病人挂号、病人确认号码。

#### （1）病人登陆



该界面是一个名为“病人登陆”的窗口。窗口标题为“病人登陆”，包含最小化、最大化和关闭按钮。窗口主体背景为浅黄色，顶部有“病人登陆”标题。下方有两个输入框：第一个输入框标签为“病人编号:”，框内显示“000001”；第二个输入框标签为“密码:”，框内显示三个黑点。在输入框下方有两个按钮，分别是“登录”和“清除”。

该 UI 界面需要用户输入病人编号和密码，并提供登录和清除按钮，点击登陆将查询数据库 T\_BRXX 内容进行用户身份的验证，点击清除按钮将清除当前光标所在控件（病人编号输入框或密码输入框）的内容。

## (2) 病人挂号

病人身份验证成功后，登陆界面将会关闭，进入病人挂号界面。



门诊挂号

科室名称 000002 外科 WK 医生名称 汪道文 WDW

号码类别 普通号 PTH 号码名称 外科普通号 WKPT

交款金额 2.0 应缴金额 1.0

找零金额 1.0 挂号号码

确定 清除 退出

在该 UI 界面中，用户选择或手动输入拼音字首过滤选择科室名称、医生名称、号码类别、号码名称。一旦选择科室，医生列表仅显示该科室的医生，即仅可选择该科室的医生；一旦选中医生，若该医生为专家医生，则号码类别中有专家号和普通号可供选择，若为普通医生，则仅有普通号可供选择；一旦选择号种类别，则仅有对应号种的号种名称可供选择，实现层层信息过滤，方便用户操作和选择。

UI 根据当前选择具体号种，自动填充该号种的应缴金额，并获取数据库中当前病人的预存金额，若足够支付选中号种的挂号费用，则无需输入交款金额和找零金额，此时交款金额和找零金额的输入控件处于不可写状态。若不足够支付当前费用，病人需要输入一定的交款金额，UI 计算后自动填充找零金额的内容。

当除挂号号码以外的所有控件的信息均已填写完整后，点击确定按钮。通过系统竞争获取挂号编号，若该号种当天的挂号人次已达到限定人数，则挂号失败，弹出当前该号种人数已满的提示框。若人数未滿，则弹出病人确定号码的倒计时窗口。

### (3) 病人确定号码



该界面用于提示用户确定抢到的号码并缴费，右上角的倒计时限制该窗口的时间，若在 10s 内未点击确定按钮确定号码并缴费，则该号码自动作废。若直接点击退出按钮，该号码也自动作废。自动作废的号码在表格 T\_GHXX 中退号标志字段设置为 true。若在 10s 内点击确定按钮，该号码有效，表格中 T\_GHXX 中退号标志位 false，并且扣除用户的费用。

注意事项：病人的客户端程序应保证以下数据一致性。

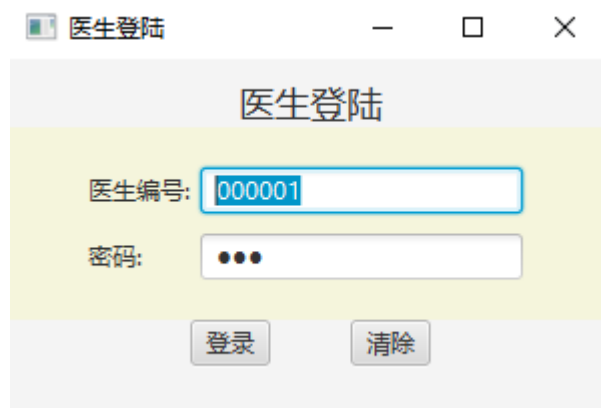
- a. 当天挂号总人数=当天各号种的挂号人次之和
- b. 病人的总支出=医生的总收入
- c. 已退号码不得用于重新挂号
- d. 所有挂号的挂号编号和挂号人次应连续不间断

### 3、设计医生系统的用户界面和功能

医生系统的用户界面可以划分为三个：医生登陆、医生主界面、医生病人列表、医生收入列表。

#### (1) 医生登陆

该 UI 界面需要医生输入医生编号和密码，并提供登录和清除按钮，点击登陆将查询数据库 T\_KSYS 内容进行用户身份的验证，点击清除按钮将清除当前光标所在控件（医生编号输入框或密码输入框）的内容。



#### (2) 医生主界面

医生身份验证成功后，登陆界面将会关闭，进入医生系统主界面。

医生主界面通过查询数据库中 T\_KSYS 信息获取并显示当前医生的基本信息，包括医生编号、医生姓名、科室名称、医生类别和最近一次的登录时间。



另外，医生主界面提供病人列表、收入列表和退出系统三个按钮。点击病人列表则弹出病人列表的窗口，显示当前医生的挂号病人。点击收入列表弹出所有科室不同医生不同号种起止日期内的收入合计的列表。点击退出系统，则该主界面和其余未关闭的病人列表界面、收入列表界面都将会被关闭。

### (3) 病人列表

点击病人列表，查询数据库中 T\_GHXX、T\_BRXX 和 T\_HZXX 弹出当前医生的挂号病人列表如下。





病人列表中，按照挂号编号升序的顺序，显示该医生病人的挂号列表、病人名称、挂号日期时间、号种名称和号种类别。该 UI 界面提供刷新按钮，可刷新当前病人列表，显示新挂号的病人。

### （3）收入列表

点击收入列表，查询数据库 T\_GHXX、T\_KSYS、T\_HZXX、T\_KSXX，弹出所有科室不同医生不同号种起止日期内的收入合计列表如下。

病人列表

医生的收入列表

起始时间: 18-5-21 上午12:00 结束时间: 18-5-21 下午12:57 查询

科室名称	医生编号	医生名称	号种名称	号种类别	挂号人次	收入合计
外科	000001	汪道文	外科普通号	普通号	4	4
外科	000001	汪道文	外科专家号	专家号	2	4

该 UI 界面提供选择起始时间和结束时间，起始时间默认为当前零时（上午 12: 00），截止日期至当前时间为止。点击查询，若结束时间在当前时间之后，则会被自动设置为当前时间，并显示收入列表，收入列表随医生和号种的不同逐行列出，每一条信息包括科室名称、医生编号、医生名称、号种名称、号种类别、挂号人次和收入合计。

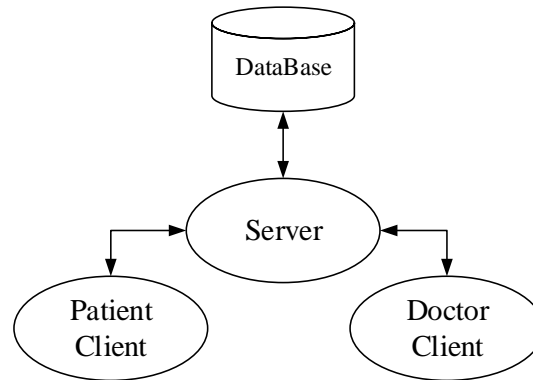
## 二、系统设计

### 1. 概要设计

为实现数据库的完整性和安全性，考虑构建服务器 Server 程序执行对数据库的查询和更新操作。因此，本次实验包含病人客户端、医生客户端和服务端三个程序，程序的基本功能和之间的交互情况如下。

服务器执行对数据库的操作，客户端向服务器发送查询信息的请求并处理来自服务器的查询结果，最终将结果映射到 UI 界面中，以图形界面的形式和用户进行信息交互。

- 1、建立并初始化数据库中基本表的信息。
- 2、维护用户状态列表，包括用户token和状态。
- 3、解析客户端传来的查询数据，构造SQL语句执行对数据库的查询或更新。
- 4、构建回复信息，将结果传送至客户端。



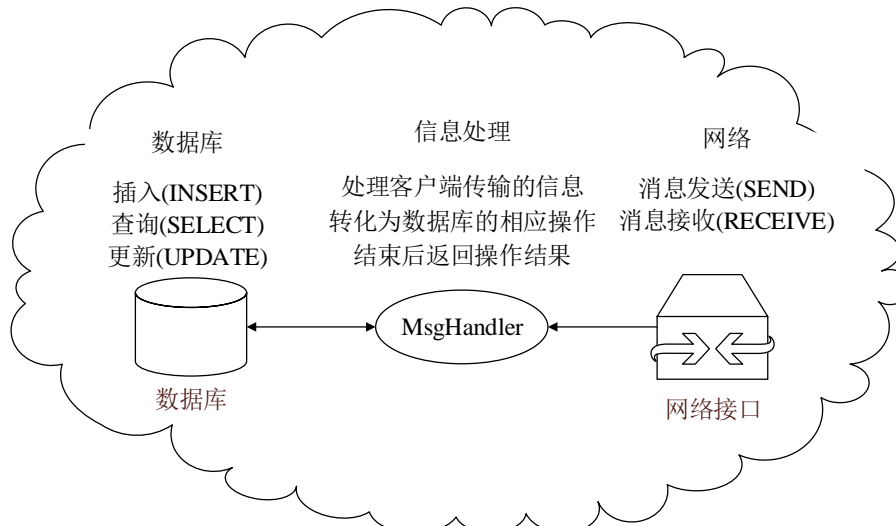
- 1、病人登陆，验证密码，若首次登陆则分发Token，此后信息传输均附带token，以验证身份。
- 2、获取科室、医生、号种等信息，根据病人预存金额判断是否需要缴纳金额，信息选择完毕后传输挂号信息，获取挂号编号。
- 3、规定时间内必须确认已获编号，否则该号作废。
- 4、号码有效，扣除病人的已有金额或缴费金额，挂号结束。

- 1、医生登陆，验证密码，若首次登陆则分发Token，此后信息传输均附带token，以验证身份。
- 2、获取医生的个人信息，显示在主界面。
- 3、获取医生的挂号病人列表。
- 4、选择指定时间段，显示所有科室不同医生不同号种起止日期内的收入合计。

下面单独介绍各个程序的结构。

## 1、服务器端

服务器端的结构如下。由于 **Server** 主要负责操作数据库，因此无需设计 **UI** 界面。服务器端开启固定的端口，用于监听来自客户端的连接，以进行消息的传输。衔接网络和数据库的结构为 **MsgHandler**，该结构解析来自客户端的信息，将其转化为 **SQL** 语句并执行，执行结果同样经 **MsgHandler** 包装成为在网络上传输的消息格式，调用网络接口发送至客户端。



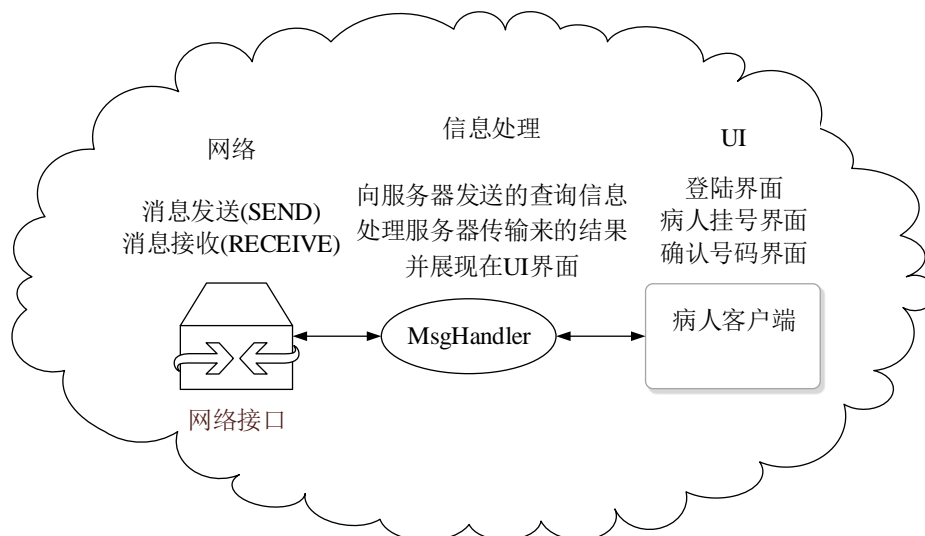
为提高执行效率，考虑将服务器端设计为多线程模式，为每个新建立的连接开启一个线程进行消息处理，以避免服务器无法处理并发的消息查询请求，导致较长时延。并且使用 TCP 短连接的模式，每次消息查询完成后就关闭当前连接，便于维护，为保证客户端身份的正确性，除登陆外，所有会话都要附带服务器端分发给客户端的 **token**，以验证身份。

由于每个查询请求逻辑较为简单，线程执行时间较短。倘若服务器端进行大量的线程创建和销毁操作，则真正用于处理消息逻辑的时间很短，效率较低，因此设计一线程池，初始状态下线程池中有一定数量的等待线程，每次有新的连接到来时，从线程池中选择一等待线程，唤醒该线程，进行消息的处理。

考虑到服务器端和客户端可能时间不一致，因此所有的登录时间或挂号时间均以服务器时间为准。

## 2、病人客户端

病人客户端的结构如下。病人客户端通过 UI 界面与用户进行信息交互，用户输入或选择所需的信息，例如登陆账号、登陆密码、挂号科室、挂号医生等，UI 界面将其获取并传递给 **MsgHandler**，**MsgHandler** 负责封装 UI 传来的信息，通过网络接口将查询请求发送服务器端并等待服务器端的返回信息。收到服务器端的响应后，客户端解析返回的信息，解封装后传递至 UI 层，将结果显示在界面上，以方便用户的下一步操作。



病人客户端首先需要用户登陆，登陆时输入病人编号及密码，若密码成功，则服务器端分发 **token** 给客户端。每一时刻每个用户只允许登陆一个客户端，因此若当前该用户已在其他客户端登陆，则此时该客户端上该用户登录失败。

病人的主要挂号界面如下。病人处于登录状态，可以选择科室、号种和医生（非专家医生不得挂专家号，专家医生可以挂普通号），对于前四项，用户可以选择输入，也可以输入信息拼音首字母输入。考虑到消息传输的最少原则，对控件设置监听器，例如只有在科室选择完毕后，才会向服务器端发送获取该科室医生的请求。前四项选择完毕后，自动填充应缴金额为该号种所需要的挂号费用，并查询该病人的预存金额是否足够缴纳该号种的挂号费用，若满足则自动设置交款金额和找零金额的控件为不可编辑，反之则需要用户输入一定金额以交款，输入完成后，系统计算并显示找零金额。点击确认获取挂号编号，规定必须在 **10s** 内

确定缴费，若超时或不点击确定，则该编号作废，且该号码不用于重新挂号。另外，每种号种均限定当日的挂号人次，若对于某一号种，当前已达到限定人次，则挂号失败。



门诊挂号

科室名称: 000002 外科 WK 医生名称: 汪道文 WDW

号码类别: 普通号 PTH 号码名称: 外科普通号 WKPT

交款金额: 余额充足 无需充值 应缴金额: 1.0

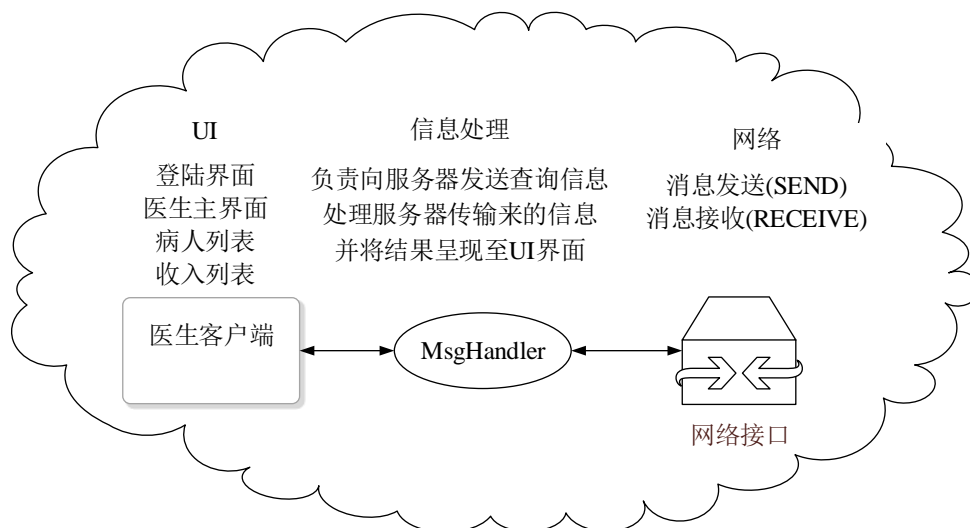
找零金额: 余额充足 无找零 挂号号码: 000018

确定 清除 退出

清除按钮负责清空当前光标所在控件的内容，退出按钮负责关闭所有窗口并向服务器端发送该病人下线信息，服务器端验证用户身份后将其状态设置为下线。

### 3、医生客户端

与病人客户端类似的，医生客户端也包含 UI 界面、消息处理层和网络接口三个部分。医生客户端呈现的 UI 界面和提供的服务有医生登录、医生主界面、查看病人列表和收入列表。



对于医生的登陆，同样仅允许每个用户在一个客户端上登陆。验证编号和密码正确后，服务器端分发 token 至客户端，客户端进入医生主界面，该界面内容如下。

界面显示医生的编号、名称、所属科室、医生类别和最近登陆时间，并且提供按钮“病人列表”和“收入列表”，供医生查看自己的挂号病人列表，所有科室不同医生不同号种的收入合计。



病人列表显示当前医生的病人的挂号信息，包括挂号编号、病人名称、挂号时间、号种名称和号种类别。医生可以点击刷新按钮，刷新当前的病人列表，以获取最新挂号信息。

收入列表显示所有科室不同医生的不同号种的收入，选择起始时间和结束时间，查询指定时间段的信息，包括科室名称、医生编号、医生名称、号种名称、号种类别、挂号人次和收入合计。

综上可知，该程序可以划分为网络接口、MsgHandler、数据库和 UI 界面四个模块，下面依次介绍各模块的功能。

### 1、网络接口

网络接口出现在客户端和服务端，实现客户端和服务端之间信息的网络传输。该模块主要实现在 **ServerNet** 类（服务器端）/**DclientNet** 类（医生客户端）/**PclientNet**（病人客户端）中。

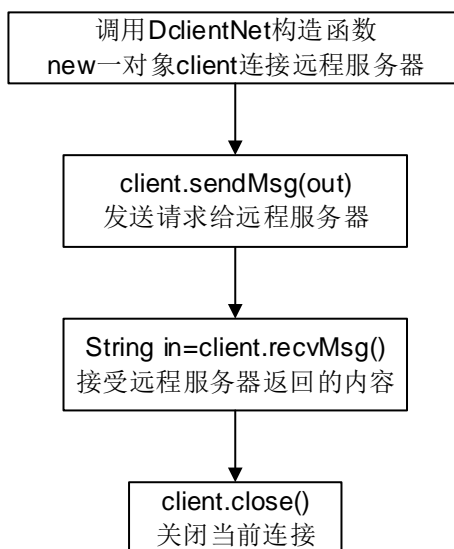
（1）客户端的 **ClientNet** 类中的方法和功能如下（以医生客户端 **DclientNet** 为例）。

DClientNet
client : Socket
port : int
serverName : String
+ DClientNet()
+ sendMsg(String) : void
+ recMsg() :String
+ closeNet() : void

网络传输利用 **Socket** 实现，**serverName** 为服务器所在主机的 IP 地址，**port** 为连接的服务器的端口。**Socket** 为当前短连接通信所用的套接字的引用。

在 **DclientNet** 构造函数中，利用 **serverName** 和 **port** 初始化 **client** 引用，实现与远程主机的连接。在 **sendMsg** 函数中，实现向远程主机发送字符串类型的数据，在 **recMsg** 函数中实现从远程主机接受字符串，并作为返回值返回，**closeNet** 函数关闭当前套接字。

因此，若要在客户端实现短连接，该类的使用过程如下。



客户端主动发起向服务器的连接，而服务器端则是在固定端口监听等待来自客户端的连接，并多线程处理客户端的请求。

(2) 服务器端的 **ServerNet** 类中的方法和功能如下。

**ServerNet** 工作的基本流程为，服务器在固定端口监听客户端的连接，每当有一个连接到来，服务器就会从线程池中唤醒一等待线程，建立与客户端的连接并处理信息查询的请求。

ServerNet
-server : Socket
-port : int
-serverSocket : ServerSocket
-corePoolSize : int
-maximumPoolSize : int
-maximumPoolSize : int
-keepAliveTime : long
-threadPoolExecutor : ThreadPoolExecutor
-workQueue : ArrayBlockingQueue<Runnable>
+ ServerNet()
+ start() : void
+ PoolClosed() : void

MyRunnable
server : Socket
+MyRunnable(Socket)
+run : void
+sendMessage(String) : void
+ recvMsg() :String
+ closeNet() : void

**ServerNet** 构造函数初始化 **serverSocket** 和 **threadPoolExecutor**，服务器端在 **port** 端口利用 **serverSocket** 监听来自客户端的连接请求，**threadPoolExecutor** 实现线程池的初始化。

**start** 函数调用 **serverSocket.accept()** 开启监听，若有连接，其返回值赋给 **Socket** 类型的引用 **server**，以 **server** 为参数，创建 **MyRunnable** 类型的对象，调用 **threadPoolExecutor.execute** 执行 **MyRunnable** 的 **run** 函数，处理来自客户端的请求。

**MyRunnable** 类实现 **Runnable** 接口，实现 **run** 函数，在该函数中调用 **recvMsg** 接受来自客户端的查询请求，调用 **MsgHandler** 接口获取查询结果，把查询结果通过 **sendMessage** 发送



至客户端，最后调用 `closeNet` 关闭连接，该线程执行完毕。

```
class MyRunnable implements Runnable{
    private Socket server;
    public MyRunnable(Socket server){
        this.server = server;
    }
    public void run(){
        System.out.println("远程主机地址: " + server.getRemoteSocketAddress());
        //应该开启一个新的线程
        String in_string = recvMsg();
        String out_string = MsgType.MsgHandler(in_string);
        sendMsg(out_string);
        closeNet();
        System.out.println("线程执行完毕");
    }
}
```

### (3) 网络信息传输格式

为实现信息传输的可扩展性，考虑使用 json 作为消息传输的格式。利用 gson 的库，实现对象和 json 字符串的自由转换。

消息传输的格式如下。

```
class Msg{
    public List<String> object;
    String token;
    int msgType;

    public Msg(int msgType){
        object = new ArrayList<String>();
        this.msgType = msgType;
    }
    public Msg(String token, int msgType){
        object = new ArrayList<String>();
        this.token = token;
        this.msgType = msgType;
    }
}
```

`token` 为客户端和服务端通信的身份凭证。`MsgType` 定义了当前消息的类型，比如病人登陆、医生登陆等。`object` 则是需要相互传输的信息，例如病人编号、密码等，若对应某一对象，例如病人，也可考虑 `object` 存在病人信息对应的 json 字符串。举例如下：

```
{"object":["{"BRBH":\\"000001\\",\\"DLKL":\\"BR1\\"}],\\"msgType":1}
{"object":["{"isOK":true,\\"token":\\"z}A\\u0014w\\u001d\\r\\o\\G\\#x\\K\\u0005E\\0\\0\\0\\",\\"reason":0}],\\"msgType":2}
```

第一个字符串为客户端传输病人编号、登录口令至服务器，目的是登陆，此时未分发

token，因此 token 字段不存在。第二个字符串为服务器端的响应结果，isOK 为 true 说明验证成功，服务器分发 token。

对象 Object 和 json 字符串之间的相互转换主要由如下两个函数实现。

```
public static String objectToJson(Object object) {
    return googleJson.toJson(object);
}

public static <T> Object jsonToObject(String json, Class<T> class1) {
    return googleJson.fromJson(json, class1);
}
```

## 2、MsgHandler

MsgHandler 实现服务器端和客户端消息的处理。MsgHandler 定义了一些消息类型和一个消息处理的方法，该方法对于不同的消息类型采取不同的处理措施，另外 MsgHandler 中还存在一 token 字段，在首次登陆成功后赋初值，此后的消息传输均需要该字段。

### (1) 客户端

客户端的 MsgHandler 主要由 UI 调用，UI 将一些界面参数和所需要的消息类型传递给 MsgHandler，获取其返回值反映到 UI 中。

对于医生客户端，存在如下消息类型，分别表示医生登陆、医生身份验证、错误编号（编号不存在）、口令错误、用户已上线、查询当前医生基本信息、医生下线、医生的病人列表和医生的收入列表。

```
public final static int DoctorLogin = 21;
public final static int DoctorLoginVerify = 22;
public final static int DoctorWrongYSBH = 23;
public final static int DoctorWrongDLKL = 24;
public final static int DoctorAlreadyOn = 25;
public final static int QueryDoctor = 26;
public final static int DoctorLogout = 27;
public final static int PatientList = 28;
public final static int SalaryList = 29;
```

对于病人客户端，存在如下消息类型，分别表示病人登陆、病人身份验证、错误编号（编

```
public final static int PatientLogin = 1;
public final static int PatientLoginVerify = 2;
public final static int WrongBRBH = 3;
public final static int WrongDLKL = 4;
public final static int QueryKSXX = 5;
public final static int QueryYSXX = 6;
public final static int QueryHZXX = 7;
public final static int QueryYCJE = 8;
public final static int PatientLogout = 9;
public final static int PatientAlreadyOn = 10;
public final static int CutPatientMonty = 11;
public final static int NumberQuery = 12;
public final static int NumberGiveup = 13;
```



号不存在)、口令错误、查询科室信息、查询医生信息、查询号种信息、查询预存金额、病人下线、该病人已登录、扣除病人金额、挂号号码查询、挂号号码放弃。

服务器端包含以上两个客户端的所有消息类型,解析来自客户端的消息,转化为数据库查询语句,执行数据库查询。

### 3、数据库

数据库的创建、查询、更新均在服务器端进行,将对数据库的操作封装成为如下类。

ServerDB
-connection : Connection
-statement : Statement
-resultSet : ResultSet
+ InitDB() : void
+ InitBRXX() : void
+ InitKSXX() : void
+ InitKSYS() : void
+ InitHZXX() : void
+ InitGHXX() : void
+ sqlExecute(String) : ResultSet
+ sqlExit() : void

该类中 InitDB 调用 InitBRXX、InitKSXX、InitKSYS、InitHZXX、InitGHXX 这五个函数,建立基本表并插入初始数据。sqlExecute 执行传入的 sql 语句,并将结果 resultSet 作为返回值返回,该函数对 UPDATE 和 SELECT 等不同类型的 sql 语句调用不同的底层函数进行执行。为防止数据库被锁,在每次执行完对数据库操作并获取到相应数据后,应当调用 sqlExit 关闭数据库。

```
public static ResultSet sqlExecute(String sql) {
    try {
        connection = null;
        statement = null;
        resultSet = null;
        Class.forName("org.sqlite.JDBC");
        connection = DriverManager.getConnection( url: "jdbc:sqlite:server.db");
        connection.setAutoCommit(false);

        statement = connection.createStatement();
        if (sql.contains("UPDATE") || sql.contains("INSERT") || sql.contains("DELETE")) {
            statement.executeUpdate(sql);
            connection.commit();
        }
        else if(sql.contains("SELECT")){
            resultSet = statement.executeQuery(sql);
        }
        else if(sql.contains("TRANSACTION")){
            statement.execute(sql);
        }
    }
}
```

#### 4、UI 界面

UI 界面利用 `javafx`，操作相应控件，并设置监听器，实现关联操作。

## 2. 详细设计

下面首先介绍几个类，并详细介绍 `MsgHandler` 函数。

### 1、用户类

#### (1) 病人

BR
+ BRBH : String
+ BRMC : String
+ DLKL : String
+ DLRQ : Date
+ YJCE : Double
+ BR(String, String, String, Date, Double)

BRVerify
+ isOK : Boolean
token : String
+ reason : int
+ BRVerify()
+ BRVerify(Boolean, String, int)

病人包括病人的基本信息和病人登陆时的验证信息，`BR` 类中的数据成员和数据库中 `T_BRXX` 的属性列保持一致，`BRVerify` 包含服务器分发给客户端的 `token`、验证是否成功和验证不成功时的原因。病人类和病人验证类为创建 `json` 字符串而设置。

#### (2) 医生

YS
+ YSBH : String
+ KSBH : String
+ YSMC : String
+ PYZS : String
+ DLKL : String
+ SFZJ : Boolean
+ DLRQ : Date
+ YS(String, String, String, String, String, String, Boolean, DLRQ)

YSVerify
+ isOK : Boolean
token : String
+ reason : int
+ YSVerify()
+ YSVerify(Boolean, String, int)

医生同样包含医生类和医生验证类。医生类和医生验证类为创建 `json` 字符串而设置。

另外由于医生客户端需要查询医生的收入信息，建立 `YS_Table` 类，对应收入信息的各列表。

YS_Table
+ KSMC: String
+ YSBH : String
+ YSMC : String
+ HZMC : String
+ HZLB : Boolean
+ GHRC : Integer
+ SRHJ : Integer
+ YS_Table (String, String, String, String, Boolean, Integer, Integer)

### (3) 号种

号种信息对应数据库中的 T\_HZXX 表的属性列。号种类为创建 json 字符串和数据库操作而设置。

HZ
+ HZBH : String
+ HZMC : String
+ PYZS : String
+ KSBH : String
+ SFZJ : Boolean
+ GHFY : Date
+ HZ (String, String, String, String, Boolean, Date)

### (4) 科室

科室类和 T\_KSXX 的属性列对应。科室类为创建 json 字符串和数据库操作而设置。

KS
+ KSBH : String
+ KSMC : String
+ PYZS : String
+ KS (String, String, String)

### (5) 挂号（仅病人客户端和服务端有该类）

GH
+ GHBH : String
+ HZBH : String
+ YSBH : String
+ BRBH : String
+ GHRC : Integer
+ SFZJ : Boolean
+ GHFY : Double
+ RQSJ : Date
+ GH (String, String, String, String, Integer, Boolean, Double, Date)

挂号类和 T\_GHXX 的属性列对应。挂号类为创建 json 字符串和数据库操作而设置。

另外由于医生客户端需要查询病人的挂号信息，建立 GH\_Table 类，便于向表中插入数据。

GH_Table
+GHRQ : Date
+ GHBH : String
+ BRMC : String
+ HZMC : String
+ SFZJ : Boolean
+ GH_Table (String, String, String, Boolean, Date)

#### (6) Token

Token
- userMap : Map<String, User>
+ putData(String, User) : void
+ isValid(String) : Boolean
+ getUser(String) : User
+ hasUser(String, String) : Boolean
+ hasUser(String, Person) : User

Token 类用于维护所有用户（医生/病人）的 token 列表，主要数据成员为 userMap，为 token-User 对，以 token 为索引，User 类的内容如下，包含了用户的状态、编号和类型。

```
enum State{
    ON, OFF
}
enum Person {
    Patient, Doctor
}
class User {
    Person person;
    String bh;
    State state;

    public User(Person person, String bh, State state) {
        this.person = person;
        this.bh = bh;
        this.state = state;
    }

    public String getToken() {
```

putData 向 map 中添加用户信息, isValid 检验传入参数 token 的有效性, getUser 利用传入 token 查询对应用户, hasUser 判断是否有当前用户, 传入 token 和编号。重载的 hasUser 返回指定类型和编号的用户。

## 2、MsgHandler 函数

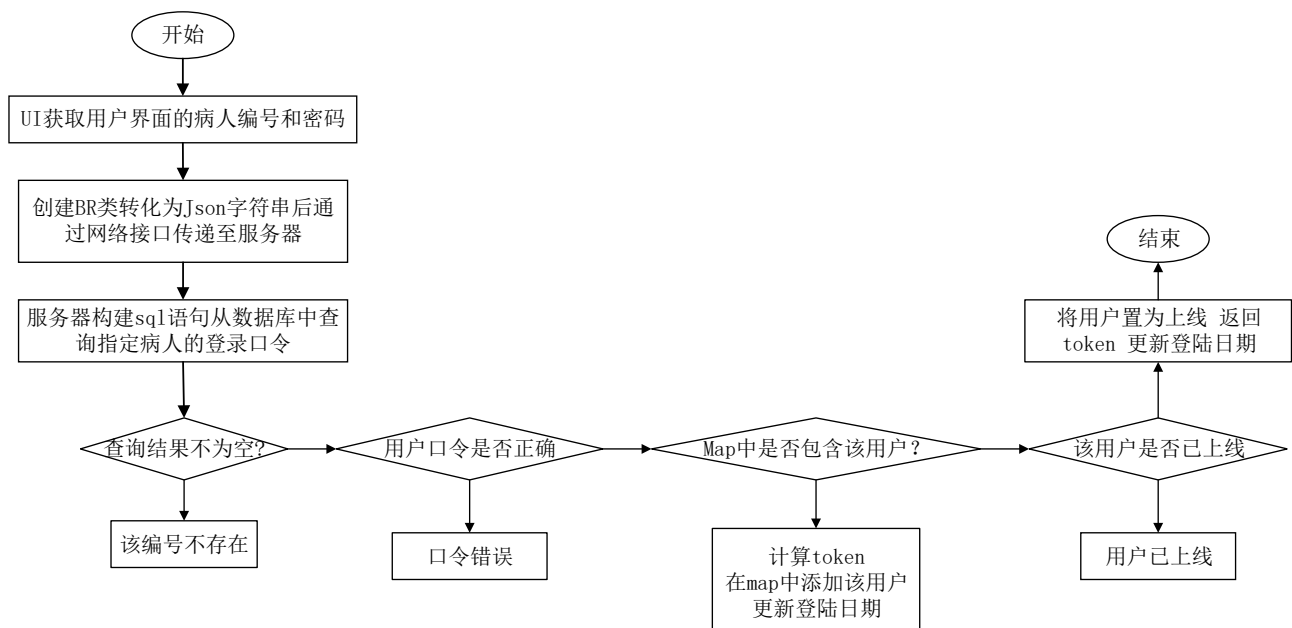
MsgHandler 的处理各消息类型的过程如下。

### (1) Login (医生登陆过程类似)

调用时期: 病人登陆

入口参数: 病人类的 json 字符串, 包含病人病人编号和登录口令。

出口参数: 验证结果, 若正确则返回 true, 否则返回错误原因, 如编号不存在、口令错误、用户已登陆。



流程：客户端通过服务器端返回的信息，弹出该医生编号不存在、口令错误该用户已上线等提示，或验证正确进入挂号界面。

### （2）QueryKSXX

调用时期：病人进入挂号界面，发送查询科室信息的请求

入口参数：无，只需发送用户 token 和类型为 QueryKSXX 的消息即可

出口参数：所有科室的信息，以 List<KS>的形式返回值 UI。

流程：服务器端接收到来自客户端的查询科室信息的请求后，首先验证 token 是否有效，若无效直接返回空字符串。若 token 有效，则构造 Sql 语句查询 T\_KSXX 表，利用查询结果构造 KS 类型的对象，将其转化为 json 字符串逐个添加至消息类型的 object 成员变量中，最后将消息转化为 json 字符串返回至客户端。客户端利用 object 中的字符串逐个还原 KS 对象，添加至 List<KS>中返回至 UI。UI 根据该 list 设置科室名称这个下拉框的选项，供用户选择。

### （3）QueryYSXX

调用时期：病人选择科室后，发送查询该科室医生的请求

入口参数：病人当前选择的科室，并发送带有用户 token 和类型为 QueryYSXX 的消息。

出口参数：指定科室所有医生的信息，以 List<YS>的形式返回值 UI。

流程：服务器端接收到来自客户端的查询医生信息的请求后，首先验证 token 是否有效，若无效直接返回空字符串。若 token 有效，则构造 Sql 语句查询 T\_KSYS 表中 KSBH 为指定科室的医生，利用查询结果构造 YS 类型的对象，将其转化为 json 字符串逐个添加至消息类型的 object 成员变量中，最后将消息转化为 json 字符串返回至客户端。客户端利用 object 中的字符串逐个还原 YS 对象，添加至 List<YS>中返回至 UI。UI 根据该 list 设置医生名称这个下拉框的选项，供用户选择。

### （4）QueryHZXX

调用时期：病人选择号种类别后，发送查询指定类别的号种名称的请求

入口参数：UI 根据当前选择科室、号种类别，构造 HZ 类型的对象，其中需要查询的字段为空，将其转化为 json 字符串传递至 MsgHandler。

出口参数：指定号种类别、科室的所有号种的信息，以 List<HZ>的形式返回值 UI。

流程：服务器端接收到来自客户端的查询号种信息的请求后，首先验证 token 是否有效，若无效直接返回空字符串。若 token 有效，则构造 Sql 语句查询 T\_HZXX 表中 KSBH 为指定科室的医生且 SFZJ 匹配的元组，利用查询结果构造 HZ 类型的对象，填充需要查询的字段例如号种名称，将其转化为 json 字符串逐个添加至消息类型的 object 成员变量中，最后将消息转化为 json 字符串返回至客户端。客户端利用 object 中的字符串逐个还原 HZ 对象，添加至 List<HZ>中返回至 UI。UI 根据该 list 设置号种名称这个下拉框的选项，供用户选择。

### （5）QueryYCJE

调用时期：病人所有有关号种的信息均已选择完毕后，查询预存金额，以判断用户是否需要缴费。

入口参数：UI 传入包含当前病人信息的 json 字符串，查询该用户当前的预存金额。

出口参数：带有当前用户预存金额的 **BR** 类构成的 **json** 字符串。

流程：服务器端接收到来自客户端的查询预存金额的请求后，首先验证 **token** 是否有效，若无效直接返回空字符串。若 **token** 有效，则构造 **Sql** 语句查询 **T\_BRXX** 表中 **BRBH** 为指定编号的病人，利用查询结果构造 **BR** 类型的对象，填充需要查询的字段即预存金额，将其转化为 **json** 字符串添加至消息类型的 **object** 成员变量中，最后将消息转化为 **json** 字符串返回至客户端。客户端利用 **object** 中的字符串还原 **BR** 对象，并返回病人的预存金额至 **UI**，以设置应缴金额和找零金额是否可编辑。

### （6）Logout（医生下线过程类似）

调用时期：病人关闭挂号窗口或点击退出按钮后，向服务器发送下线请求。

入口参数：**UI** 传入包含当前病人信息的 **json** 字符串，向服务器说明该病人即将下线。

出口参数：服务器的处理结果，正确下线或错误下线。

流程：服务器端接收到来自客户端的查询号种信息的请求后，首先验证 **token** 是否有效，若无效直接返回空字符串。若 **token** 有效，在 **map** 表中查询该用户，若存在该用户，则将该用户的状态设置为 **OFF**，表示用户下线，返回客户端操作结果 **true**。否则返回 **false** 表示下线错误。

### （7）CutPatientMoney

调用时期：病人确定系统抢到的挂号编号并确定缴费后，客户端发送扣除用户费用的请求。

入口参数：将要扣除的金额，由当前选择号种和用户的预存金额有关。

出口参数：扣除金额的操作结果。

流程：服务器端接收到来自客户端的扣除用户金额信息的请求后，首先验证 **token** 是否有效，若无效直接返回空字符串。若 **token** 有效，在 **map** 表中查询该用户，并调用一加有同步锁的函数，实现金额的扣除并且不会出现线程竞争导致结果错误的后果。

在加同步锁的函数中，首先从数据库中获取病人预存金额，将其减去当前应扣费金额后，再更新 **T\_BRXX** 中用户的预存金额。若不加同步锁，则可能导致用户支出和医院收入不匹配。

### （8）NumberQuery

调用时期：病人所有挂号信息填写完整后，点击确定按钮进行系统抢号时调用。

入口参数：将要挂的号码的信息，即 **GH** 类型的对象，以 **json** 字符串的形式传递给 **MsgHandler**。

出口参数：系统竞争到的挂号编号。

流程：服务器端接收到来自客户端的查询挂号编号的请求后，首先验证 **token** 是否有效，若无效直接返回空字符串。若 **token** 有效，还原传入的 **GH** 类型的对象，调用一加有同步锁的函数，实现挂号信息正确的插入 **T\_GHXX** 表中。

在加同步锁的函数中，首先获取新加入的信息的编号，是最后一条记录的挂号编号加一。并且查询当日该号种的挂号人次，若即将超过限制人次，则返回当前号种已满的信息。反之将新的挂号记录附上数据库的当前时间插入到 **T\_GHXX** 表中。由于该过程存在读取、加一再

写入数据库的过程，若不加同步锁，则会导致挂号号码不连续的后果。

#### (9) NumberGiveup

调用时期：病人放弃抢到号码时调用。

入口参数：将要退号的号码信息，即 GH 类型的对象，以 json 字符串的形式传递给 MsgHandler。

出口参数：操作结果。

流程：服务器端接收到来自客户端的放弃挂号的请求后，首先验证 token 是否有效，若无效直接返回空字符串。若 token 有效，还原传入的 GH 类型的对象，并将数据库中该元组的退号标志设置为 true。

#### (10) QueryDoctor

调用时期：医生身份经过验证后，查询医生的相关信息时调用，以显示在医生主界面中。

入口参数：YS 类型的对象转化的 json 字符串，其中需要查询的字段为空。

出口参数：所有的查询结果。

流程：服务器端接收到来自客户端的查询医生详细信息的请求后，首先验证 token 是否有效，若无效直接返回空字符串。若 token 有效，还原传入的 YS 类型的对象，通过医生编号，将数据库 T\_KSYS 中查询医生的详细信息，在 T\_KSXX 中查询医生所在科室的信息，最后都转化为 json 字符串添加到 object 中，返回给客户端，客户端解析后将其呈现在医生主界面。

#### (11) PatientList

调用时期：医生点击主界面的病人列表按钮或点击病人列表中的刷新按钮时调用。

入口参数：YS 类型的对象转化的 json 字符串。

出口参数：存放该医生所有的病人信息的 ObservableList<GH\_Table>对象。

流程：服务器端接收到来自客户端的查询病人列表的请求后，首先验证 token 是否有效，若无效直接返回空字符串。若 token 有效，还原传入的 YS 类型的对象，通过医生编号，构造如下 SQL 语句，查询挂号信息表中为退号且医生编号为指定医生编号的记录。

```
String sql = "SELECT T_GHXX.GHBH, T_BRXX.BRMC, T_HZXX.HZMC, T_HZXX.SFZJ, T_GHXX.RQSJ " +  
    "FROM T_GHXX, T_BRXX, T_HZXX " +  
    "WHERE YSBH = '" + ys.YSBH + "' AND THBZ = 0 " +  
    "AND T_GHXX.BRBH = T_BRXX.BRBH AND T_HZXX.HZBH = T_GHXX.HZBH ";  
ResultSet resultSet = ServerDB.sqlExecute(sql);
```

利用查询结果逐个构造 GH\_Table 类型的对象，转化成为 json 字符串后添加到 object 中，作为内容返回至客户端。客户端依次还原每个 GH\_Table 对象，加入 ObservableList<GH\_Table>中，并呈现在表格上。

#### (12) SalaryList

调用时期：医生进入收入列表界面，选择起始时间和截止时间后，点击查询时调用。

入口参数：用户在 UI 界面选择的起始时间和截止时间。

出口参数：存放所有科室不同医生不同号种的收入情况的 ObservableList<YS\_Table>类



型的对象。

流程：服务器端接收到来自客户端的查询医生收入列表的请求后，首先验证 **token** 是否有效，若无效直接返回空字符串。若 **token** 有效，还原传入的起始时间和截止时间，构造如下 **SQL** 语句，查询未退号，且挂号日期处于指定范围内的记录，并且根据医生编号和号种编号的不同进行聚合，最后计算个数和总收入。

利用查询结果构造 **YS\_Table** 类型的对象，将其转化为 **json** 字符串后添加到 **object** 中，返回至客户端。客户端依次还原每个 **YS\_Table** 对象，添加至 **ObservableList<YS\_Table>** 中，并呈现在表格上。

```
String sql = "SELECT T_KSXX.KSMC, T_GHXX.YSBH, T_KSYS.YSMC, T_HZXX.HZMC, T_HZXX.SFZJ HZLB, COUNT(*) GHRC," +
    " SUM(T_GHXX.GHXY) SRHJ FROM T_GHXX, T_KSXX, T_HZXX, T_KSYS " +
    "WHERE T_GHXX.YSBH = T_KSYS.YSBH AND T_GHXX.HZBH = T_HZXX.HZBH AND T_KSYS.KSBH = T_KSXX.KSBH " +
    "AND T_GHXX.THBZ = 0 " +
    "AND T_GHXX.RQSJ BETWEEN '" + start_time.toString() + "' AND '" + end_time.toString() + "' " +
    "GROUP BY T_GHXX.YSBH, T_GHXX.HZBH";
ResultSet resultSet = ServerDB.sqlExecute(sql);
```

### 三、软件开发

开发环境：IntelliJ IDEA

JDK 版本：jdk\_1.8.0\_144

使用 lib 库：gson

数据库：sqlite

网络传输：socket

### 四、软件测试

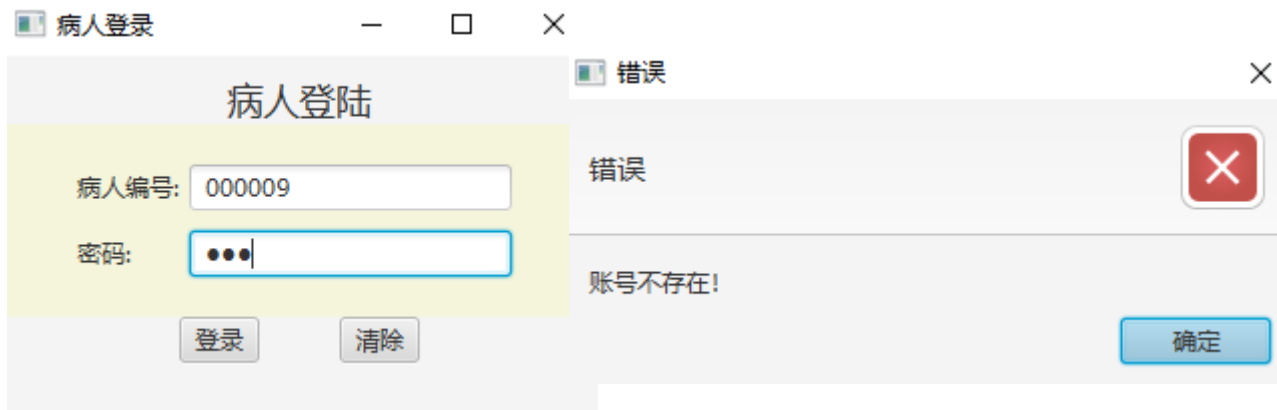
1、运行服务器，实现数据库的初始化。

服务器被初次运行时，会初始化数据库的内容。此时数据库文件 **server.db** 被创建，且其中除 **T\_GHXX** 以外，所有基本表均已添加初始数据，，例如病人信息的表格内容如下。

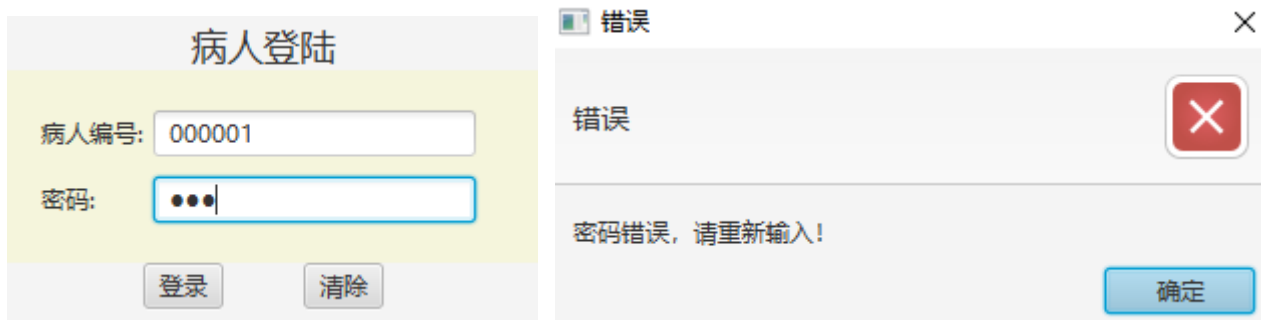
	BRBH	BRMC	DLKL	YCJE	DLRQ
1	000001	BR1	BR1	5	NULL
2	000002	BR2	BR2	2	NULL
3	000003	BR3	BR3	3	NULL
4	000004	BR4	BR4	4	NULL
5	000005	BR5	BR5	5	NULL
6	000006	BR6	BR6	6	NULL
7	000007	BR7	BR7	7	NULL
8	000008	BR8	BR8	8	NULL

2、登陆病人客户端

输入病人的编号和密码，若编号不在 **T\_BRXX** 中有记录，则弹出该用户不存在的提示框如下，例如输入编号 000009。



若当前编号存在，但密码不正确，会弹出密码不正确的提示框。



只有在编号和密码都正确的情况下，病人才能登陆成功。若点击确定时编号或密码未填写，则会在界面中显示提示未填写字段为空。



输入正确的病人编号和密码，进入挂号界面。此时若再次开启一个客户端，同样登陆 000001 号病人的系统，则会提示用户已上线的提示。



另外，在病人编号处设置了监听器，只允许输入数字字符串，不允许输入其他字符。

### 3、病人挂号

The screenshot shows a patient registration form with the following fields and values:

Field	Value
科室名称 (Department Name)	000001 内科 NK
号码类别 (Code Type)	000002 外科 WK
交款金额 (Payment Amount)	000003 耳鼻喉科 EBHK
找零金额 (Change Amount)	000004 放射科 FSK
医生名称 (Doctor Name)	医生名称
号码名称 (Code Name)	号种名称
应缴金额 (Due Amount)	
挂号号码 (Registration Number)	

The dropdown menu for the department name is open, showing a list of departments: 000001 内科 NK, 000002 外科 WK, 000003 耳鼻喉科 EBHK, 000004 放射科 FSK, 000005 口腔科 KQK, 000006 美容外科 MRWK, 000007 儿科 EK, 000008 急诊科 JZK, and 000009 妇科 FK.

病人光标点击下拉框，弹出下拉框内容，例如科室名称的信息。可以手动点击要选择的科室，也可以输入拼音字首进行过滤，如下图。

The screenshot shows the patient registration form with the department name dropdown menu open and filtered by the letter 'W'. The visible options are:

- 000002 外科 WK
- 000006 美容外科 MRWK

选择科室后，光标点击医生名称下拉框，弹出该科室的医生，同样可以手动选择和拼音首字符过滤输入。

The screenshot shows the patient registration form with the doctor name dropdown menu open and filtered by the letter 'T'. The visible options are:

- 汪道文 WDW
- 唐家荣 TGX

号码类别和号码名称同样可以利用两种方式输入。前四项选择完毕后，自动填充应缴金额为 1.0 元，同时由于病人有一定的预存金额足够支付该缴费费用，因此交款金额和找零金额被自动填充为如下内容。

The screenshot shows the patient registration form with the following fields and values:

Field	Value
科室名称 (Department Name)	000002 外科 WK
医生名称 (Doctor Name)	汪道文 WDW
号码类别 (Code Type)	普通号 PTH
号码名称 (Code Name)	外科学普通号 WKPT
交款金额 (Payment Amount)	余额充足 无需充值
应缴金额 (Due Amount)	1.0
找零金额 (Change Amount)	余额充足 无找零
挂号号码 (Registration Number)	



此时所有除挂号信息以外的内容均已填写完整，点击确定系统将竞争编号。  
若 10s 内未确认，则该编号自动作废，挂号编号被填充信息如下。

交款金额	余额充足 无需充值	应缴金额	1.0
找零金额	余额充足 无找零	挂号号码	000001已失效

此时 T\_GHXX 中有一项记录为已作废号码的记录，其中退号标志为 1。

	GHBH	HZBH	YSBH	BRBH	GHRC	THBZ	GHFY	RQSJ
1	000001	000003	000001	000001	1	1	1	Mon May 21 21:54:21 CST 2018

为测试挂号人次的限制，不断点击确定多次挂该种类的号，由于预先设置该号种的挂号人次为 3，当试图挂编号 000005 的号码时，提示该号种今日已满的信息。



此时挂号信息表中内容如下，显示该号种今日已挂号三人次。

	GHBH	HZBH	YSBH	BRBH	GHRC	THBZ	GHFY	RQSJ
1	000001	000003	000001	000001	1	1	1	Mon May 21 22:07:17 CST 2018
2	000002	000003	000001	000001	1	0	1	Mon May 21 22:07:19 CST 2018
3	000003	000003	000001	000001	2	0	1	Mon May 21 22:07:21 CST 2018
4	000004	000003	000001	000001	3	0	1	Mon May 21 22:07:22 CST 2018

	BRBH	BRMC	DLKL	YCJE	DLRQ
1	000001	BR1	BR1	0	Mon May 21 22:07:10 CST 2018
2	000002	BR2	BR2	2	NULL

选择挂别的号码，抢到编号 000005 的号码后，由于此时病人 1 的预存金额已减少为 0，此时缴款金额和找零金额变为可写状态，用户需要输入一定的金额以交款。

科室名称	000002 外科 WK	医生名称	汪道文 WDW
号码类别	专家号 ZJH	号码名称	外科专家号 WKZJ
交款金额	<input type="text"/>	应缴金额	2.0
找零金额	<input type="text"/>	挂号号码	000005

输入缴款金额后，系统自动计算找零金额，并填充至找零金额的控件中。

交款金额	4.0	应缴
找零金额	2.0	挂号

此时点击确定按钮，系统抢到编号，若确定该编号，则此时不再减少 T\_BRXX 表中 BR1 的预存金额，此后预存金额一直为 0，系统只扣除用户的缴款金额。

点击退出系统，用户下线。

查看 T\_GHXX 表中的信息如下。

	GHBH	HZBH	YSBH	BRBH	GHRC	THBZ	GHFY	RQSJ
1	000001	000003	000001	000001	1	1	1	Mon May 21 22:07:17 CST 2018
2	000002	000003	000001	000001	1	0	1	Mon May 21 22:07:19 CST 2018
3	000003	000003	000001	000001	2	0	1	Mon May 21 22:07:21 CST 2018
4	000004	000003	000001	000001	3	0	1	Mon May 21 22:07:22 CST 2018
5	000005	000004	000001	000001	1	0	2	Mon May 21 22:09:41 CST 2018
6	000006	000004	000001	000001	2	1	2	Mon May 21 22:13:06 CST 2018

由于 000001 编号的号码被退号，因此再次挂此号种时，挂号人次仍从 1 开始编号，因次挂号人次仅统计当日挂号成功的编号的人次。挂号编号和时间连续，且已退号种均已将退号标志设置为 1。

4、医生系统登陆和病人登陆流程一致，此处不再赘述。

5、医生主界面和病人列表、收入列表的查看。

医生身份验证成功后，进入主界面如下图。

医生编号: 000001

医生姓名: 汪道文

科室名称: 外科

医生类别: 专家医生

最近登陆时间: 2018年5月21日 下午10时19分53秒

病人列表

收入列表

退出系统

点击病人列表，显示该医生的病人列表如下。

汪道文医生的病人列表				
				刷新
挂号编号	病人名称	挂号日期时间	号种名称	号种类别
000002	BR1	Mon May 21 22:07:19 CST 2018	外科普通号	普通号
000003	BR1	Mon May 21 22:07:21 CST 2018	外科普通号	普通号
000004	BR1	Mon May 21 22:07:22 CST 2018	外科普通号	普通号
000005	BR1	Mon May 21 22:09:41 CST 2018	外科专家号	专家号

表中内容和数据库 T\_GHXX 中未退号且医生编号为 000001 的记录保持一致。此时以 BR2 的身份登陆病人系统。挂该医生的专家号，点击刷新按钮，出现 BR2 的挂号记录。

汪道文医生的病人列表				
				刷新
挂号编号	病人名称	挂号日期时间	号种名称	号种类别
000002	BR1	Mon May 21 22:07:19 CST 2018	外科普通号	普通号
000003	BR1	Mon May 21 22:07:21 CST 2018	外科普通号	普通号
000004	BR1	Mon May 21 22:07:22 CST 2018	外科普通号	普通号
000005	BR1	Mon May 21 22:09:41 CST 2018	外科专家号	专家号
000007	BR2	Mon May 21 22:23:20 CST 2018	外科专家号	专家号

点击收入列表，显示界面如下。

**医生的收入列表**

起始时间: 18-5-21 上午12:00  结束时间: 18-5-21 下午10:24 

科室名称	号种名称	号种类别	挂号人次	收入合计
2018-5-21				

▼ Date  
▶ Time

用户可以点击带有日历的按钮选择起始时间和结束时间，起始时间默认和当前 0 点即上午 12 点，结束时间默认为打开窗口时的时间，点击查询，显示病人的收入列表如下。

**医生的收入列表**

起始时间: 18-5-21 上午12:00  结束时间: 18-5-21 下午10:24 

科室名称	医生编号	医生名称	号种名称	号种类别	挂号人次	收入合计
外科	000001	汪道文	外科普通号	普通号	3	3
外科	000001	汪道文	外科专家号	专家号	2	4

当前对应的 T\_GHXX 内容如下，可以看出病人的支出和医生的收入是相等的。

	GHBH	HZBH	YSBH	BRBH	GHRC	THBZ	GHFY	RQSJ
1	000001	000003	000001	000001	1	1	1	Mon May 21 22:07:17 CST 2018
2	000002	000003	000001	000001	1	0	1	Mon May 21 22:07:19 CST 2018
3	000003	000003	000001	000001	2	0	1	Mon May 21 22:07:21 CST 2018
4	000004	000003	000001	000001	3	0	1	Mon May 21 22:07:22 CST 2018
5	000005	000004	000001	000001	1	0	2	Mon May 21 22:09:41 CST 2018
6	000006	000004	000001	000001	2	1	2	Mon May 21 22:13:06 CST 2018
7	000007	000004	000001	000002	2	0	2	Mon May 21 22:23:20 CST 2018

点击退出系统，医生下线。

## 五、特点与不足

### 1. 技术特点

1、编程时利用 **synchronized** 关键字，对数据库中必须一次性进行的查询、计算、更新（即写入）的操作集合加了同步锁，例如挂号时查询最后一个号的编号、对其进行+1 操作，再插入新的数据，这一系列的操作不能被中断，否则会导致号码不连续的情况，因此加上同步锁，在某一线程执行该函数时，其余线程必须等待，扣除预存金额的逻辑也需要添加同步锁。

2、设计线程池，由于每个线程只处理客户端的一个请求，因此每个线程的执行时间较短。若频繁执行线程创建和销毁操作，则实际执行效率很低。因此设置一线程池，其中有固定数量的线程，一旦有新的连接请求，就会从线程池唤醒一个线程处理请求，只有当线程池中沒有空闲状态的等待线程时，才会新创建线程，提高运行速度和效率。



### 3、设计服务器端

设计服务器端的初衷，是为了数据库操作的一致性和安全性，若不存在服务器端，则病人客户端的数据库要和医生客户端的数据库始终保持一致，难以操作。通过服务器将数据库保护起来，使得数据库操作更安全、方便。

### 4、利用 Json 传输数据

Json 是一种扩展性很强的消息传输格式，本次实验利用 gson 库，设计两个类，实现对象和 json 字符串之间的任意转换，只要想要传输某个类的信息，就可以将其转化为 json 字符串，只要想解析传来的 json 字符串，就可以将其转化为指定的类，极其方便。

## 2. 不足和改进的建议

1、结构设计的不是很合理，层次之间的粘连性太强，层次间接口界限不清晰，无法单独将某一层剥离，即无法单独将某一层抽离成为 jar 包供他人使用。

例如数据库提供给 MsgHandler 的接口就不是很方便使用，每次执行完后必须调用一次关闭，否则数据库会死锁。并且 UI 和 MsgHandler 的交互也很复杂。

2、有些操作比较赘余，并且频繁的创建对象后使用一两次便不再使用，需要频繁回收未被引用的对象。

3、UI 设计中对于监听器的使用不是很规范，导致代码冗余，也会出现一些奇怪的 bug。

## 六、过程和体会

### 1. 遇到的主要问题和解决方法

1、起初在服务器两个线程同时操作数据库时会发生死锁，后来发现由于每次数据库操作后未关闭数据库，才导致该错误。后修复该问题，实现了多程序的并发查询。

2、在对 ComboBox 控件添加根据输入内容过滤下拉框项目的个数时，由于该控件会在只有一个选项时自动将控件填充为该选项的内容，会导致最后过滤到只剩一个时会无法删除，最后通过条件判断捕捉这一异常情况，进行特殊处理后解决该问题。

3、对 UI 的更新应该在非 UI 线程来做，起初全部由 UI 线程实现，例如对用户登陆时内容的解析，导致最后各控件之间的逻辑混乱，后来改用其余小的线程负责刷新 UI，问题基本解决。

4、对于思考题，可以进行扣费时，再次查询确认应交金额再缴费，以实现缴费金额等于医生的收入金额。

### 2. 课程设计的体会

通过本次 Java 实验，深刻理解了 Javafx、数据库、多线程、网络编程等的使用方法，例如 Javafx 控件监听器的设计、数据库的查询、更新和插入、线程池的设计、Socket 编程的基本使用方法，并且通过自己设计整个程序的层次和封装所需类，了解到良好的框架或者设计模式会给编程者带来极大的方便，不至于修复 bug 很困难或者很复杂。








## 七、源码和说明

### 1. 文件清单及其功能说明

该实验源码分为三个文件夹：

#### 1、Server

该文件夹中存放服务器的代码，打开可以看到其中的数据库文件 `server.db`。

 .idea	2018/5/21 星期...	文件夹	
 out	2018/4/11 星期...	文件夹	
 src	2018/4/11 星期...	文件夹	
 server.db	2018/5/21 星期...	Data Base File	64 KB
 Server.iml	2018/4/11 星期...	IML 文件	1 KB

程序源码位于 `src` 文件夹中，可以直接在 `IDEA` 中打开 `Server` 文件夹，点击运行即可运行服务器程序。

#### 2、Pclient

该文件夹中存放病人客户端程序的源码。

#### 3、Dclient

该文件夹中存放医生客户端程序的源码。

### 2. 用户使用说明书

可以直接在 `IDEA` 中打开相应的文件夹，点击运行即可运行对应程序。

注意，一定要先开启服务器进程，再开启客户端，否则客户端会与服务器连接失败。

### 3. 源代码

源码文件过多且内容过长，不在此处列出，可直接查看文件夹中的源码内容。