

****

基于java和MySql的社区家庭医生管理系统

学生姓名 张寒星

学 号 20171309028

学 院 人工智能

专 业 信息工程

指导教师 马杰良

**二O二一 年 五 月**

**目 录**

1 前言 5

1.1 选题背景 5

1.2 选题目标和意义 5

2. 涉及的技术 5

2.1 开发平台MyEclipse 5

2.2 Java 5

2.3 MySql Database 6

2.4 自定义JDBC工具技术 7

2.5 对象工厂BeanFactory技术 8

2.6 事务Transaction技术 9

3. 系统分析 9

3.1 系统流程概述 9

3.2 系统需求分析 9

3.2.1 管理员和用户登录 9

3.2.2 用户账户的管理 10

3.2.3 居民与医生的签约 10

3.2.4 居民的预约 10

4. 系统设计 10

4.1 系统总体设计 10

4.2 功能模块设计 10

4.3 数据库设计 11

5. 系统实现 14

5.1 程序设计 14

6. 系统测试 18

6.1 测试任务 18

6.2 测试目标 18

6.3 测试方案 19

7. 总结 19

参考文献 19

致谢 20

基于java和MySql的社区家庭医生管理系统

张寒星

南京信息工程大学人工智能学院，江苏 南京 210044

摘要:

身体是革命的本钱，对革命的本钱来讲最重要的就是健康，如何保持一个健康的身体是人类长久以来永恒不变的话题，而医生又是维持身体健康不可或缺的一环。当今社会，大家基本都生活在社区中，而社区里都配置有卫生室和几名医生，医生在那里不可能随时都有病人前来体检或问诊，而居民们的问诊时间也不可能保证在同一个时间段。现如今，居民若要体检和看病，必须自行前往卫生室和医院，如何充分利用医生的空闲时间和现在我们拥有的技术从而给居民们带来便利呢？

当今社会，几乎家家都有一台计算机和几部手机，这便是我们可利用的资源，这些计算机和手机早已给我们提供了巨大的便利。从前的居民上门问诊方式给居民带来了极大的不便，就算身体不适也必须要出门到医生那里，为了解决此类问题，方便居民们的生活，同时合理利用医生的碎片化时间，特此设计一个社区家庭医生管理后台系统。

本论文主要介绍的是社区家庭医生管理系统的简介、原理、实现技术以及测试方法。首先介绍此次选题的背景、目标及意义，其次，对所使用到的技术进行逐一介绍，接着就是对系统需求的分析，主要是功能需求，然后，设计系统的总体和功能模块，以及系统的实现，再者就是对系统进行测试，论文最后即为总结、致谢和参考文献。

关键词：社区家庭医生管理；mysql数据库；JDBC

**Community Family Physician Management System Based On Java And MySql**

Zhang Hanxing

Institute of Artificial Intelligence，NUIST，Nanjing 210044，China

**Abstract**:The body is the capital of revolution, and the most important thing for the capital of revolution is health. How to maintain a healthy body has been an eternal topic for human beings for a long time, and doctors are an indispensable part of maintaining a healthy body. In today's society, everyone basically lives in the community, which is equipped with a health room and several doctors. It is impossible for doctors to have patients come for physical examination or consultation at any time, and residents' consultation time cannot be guaranteed at the same time. How can we make the most of the free time of doctors and the technology we have so that residents can benefit from the fact that they have to go to their own clinics and hospitals to get a medical check-up and see a doctor?

In today's society, almost every family has a computer and several mobile phones, which are the resources we can use. These computers and mobile phones have already provided us with great convenience. In the past, residents' door-to-door consultation brought great inconvenience to residents, who had to go to the doctor even if they were not feeling well. In order to solve this problem, facilitate residents' life, and make reasonable use of doctors' fragmented time, a background system of community family doctor management was designed.

This paper mainly introduces the introduction, principle, implementation technology and test method of the community family doctor management system. First introduced the selected topic background, objectives and significance, and secondly, to use the technologies are introduced, followed by the analysis of system requirements, the mainly functional requirements, then design the system of overall and functional modules, and the realization of the system, moreover is the testing system, paper finally is the summary, thanks, and references.

**Keywords**: community family doctor management;MySQL database;JDBC

1. 前言
   1. 选题背景

在如今的社区医生工作模式下，社区的医生们总会或多或少的余下一些碎片时间，同时，计算机技术飞速发展，各种便利的社区系统早已悄无声息地给居民们带来了极大的便利，为了继续增加计算机系统给居民带来的便利，同时为了使居民与医生的联系更加密切，同时使体检等事项更加方便，特此建立数据库维持一种新的医患关系。

社区家庭医生管理系统的实现将为更多家庭提供便捷、高效的医生随访上门服务，节约许多居民上门寻访的时间，居民可根据需求与医生签约，签约完成后可进行随访的预约，同时也将一定程度上解决医生时而忙碌时而无所事事的尴尬处境。随着如今医疗条件的网络化水平不断提高，网络医疗在人们的生活中正在扮演一个越来越重要的角色， 不止减少了居民上门的不便，同时也降低了医生工作的密集程度。

* 1. 选题目标和意义

居民和医生将可以通过此系统浏览和搜索医生信息。系统将为居民提供签约与预约上门操作，该系统主要包含的功能有：居民和医生账号的管理、居民和医生的签约以及居民对签约医生随访上门服务的预约。

受新冠疫情影响，越来越多的人开始注重身体的健康，和身心的检查。但医生们总是忙碌的时候非常忙碌，空闲的时候又过于空闲，这些空闲时间需要有效利用起来，服务居民和患者。健康是人类永恒不变的话题，保持一个健康的身体是每个人梦寐以求的事情，在这个网络飞速发展的时代，信息技术早已渗透入人们生活的方方面面，社区家庭医生管理系统的建立有助于让居民们都能拥有一个健康的身体，同时方便居民的生活，有利于社区开拓更多样化的服务。

1. 涉及的技术
   1. 开发平台MyEclipse

MyEclipse多语言编译设计环境，它是一个握持着许多独特而与众不同的优质效用的JavaEE集成工作技术编译环境，这个软件中不仅包括了全栈的编译功能，能够自动检测语法错误等，还自带调试功能，除此之外，它还可以进行类和项目的测试，还拥有发布系统以及版本控制提交git的功能。它是把我们自己的各种plugins加在原本的Eclipse程序的基础上进行开发而成，具有非常高质量的能效的集成技术开发工作环境，它的主要活跃领域有，中国移动互联网中的应用，以及Java语言的编译开发，还有Java EE的设计。使用它对JavaEE和各种Database的设计和发布以及集成应用服务器的运作都有极大的帮助。

* 1. Java开发语言

1995年5月，Java横空出世，最初的它，是太阳微系统公司推出的一种编程语言和语言环境的结合体。 最初的java并不叫java，它的名字是Oak，由詹姆斯·高斯林联合其他的工程师共同发布，其最大的特点之一，就是它面向的是对象，且这种环境可以用于需要跨系统的应用与软件的程序编程。

Java这门编程语言，主要的面向领域是对象，不但容纳了C++语言的独特的好处，而且剔除了C++里晦涩难懂的多继承概念与指针用法，可谓取其精华去其糟粕，所以Java编程的优点包括其优异的效用和易于学习上手的特点，这是java编程语言的两个主要属性。作为一个静态的面向对象编程语言的象征，Java编程语言的发展以绝佳的方式复现了它的基本理念，即它的面向对象的特性，而正因此，开发人员可以以优秀的教学与思维的方式进行较为高复杂度的程序编写。

利用try-catch异常处理机制，对于有几率出现异常的地方，将其置于try后的语句当中，当try语句块中的程序流程执行过程中发现异常时，可以对其中的异常进行捕捉以及处理。

若是 try 语句内的代码执行过程中出现了某些异常，那么一个被预先定义的相应异常就会被拋出，此异常作为一个对象被封装在java类中，然后 catch 中放置对异常的捕获和处理就会依据所拋出异常研究对象的类型可以进行捕获与分析，并解决。解决之后，程序将跳过try语句块中其他的语句，不再执行，跳转到catch语句块之中的第一条可执行语句，从跳转到的语句开始向后执行。

若 try 语句块中未检测到异常发生，未捕获到异常对象，那么 将会按从上到下的方式运行try 块中的语句，后面的 catch 语句块中的语句则会直接被跳过，运行完毕后，直接跳出try-catch，后续将从catch语句块之后的第一条语句即是跳出后开始执行的地方。

* 1. MySql Database

MySQL数据库，一开始是属于甲骨文公司的开源免费产品，之后此数据库技术被oracle公司合并，但仍保持着开源免费的特点。MySQL是当今世上应用最为广泛的进行整个中小型系统信息管理的关系型数据库技术中能够跻身前列的存在，在研究管各种管理系统的信息管理方面，MySQL是极佳的关系型数据库资源管理控制系统，是企业分析所较为频繁使用的技术之一。

MySQL 是一个关系数据库管理系统，其最大的优势便是它的工作模式是将信息置于使用者创建的许多张表中，取代传统的所有信息集中存放的模式，增加了可用性和效率。

MySQL中的基本语句最常用的领域是对后台数据库进行访问的数据统一化管理。MySQL主要所采用的政策是双重许可，共两种版本，分别是社区用的版本和商业用的版本。由于它比较快速，整体所拥有的成本较为低廉，规模比较小，特别是它具有的一个最大特点是源码公开，因此它通常作为非大型系统开发的数据库而被大家所选择。

此系统除了最基础的增删改查语句之外，还需要用到MySql的一个叫做存储过程的功能，可以将很多条sql语句放入这个功能中，它相当于一个对象，方便于使用call()语句进行外部调用对象，此对象存在于数据库之中。可以将多个sql语句存入存储过程之中，存储过程的调用过程中，其中的语句会自上而下先后运作，创建的方法为

create procedure+[存储过程名](参数)

begin

sql语句;

sql语句;

…

end

在存储过程中，每条sql语句都需要以分号结尾，表明这条sql语句结束，所以需要先将My Sql的结束符号进行修改，修改方法为

delimiter //

此时sql语句的结束符号就变成了两条斜杠

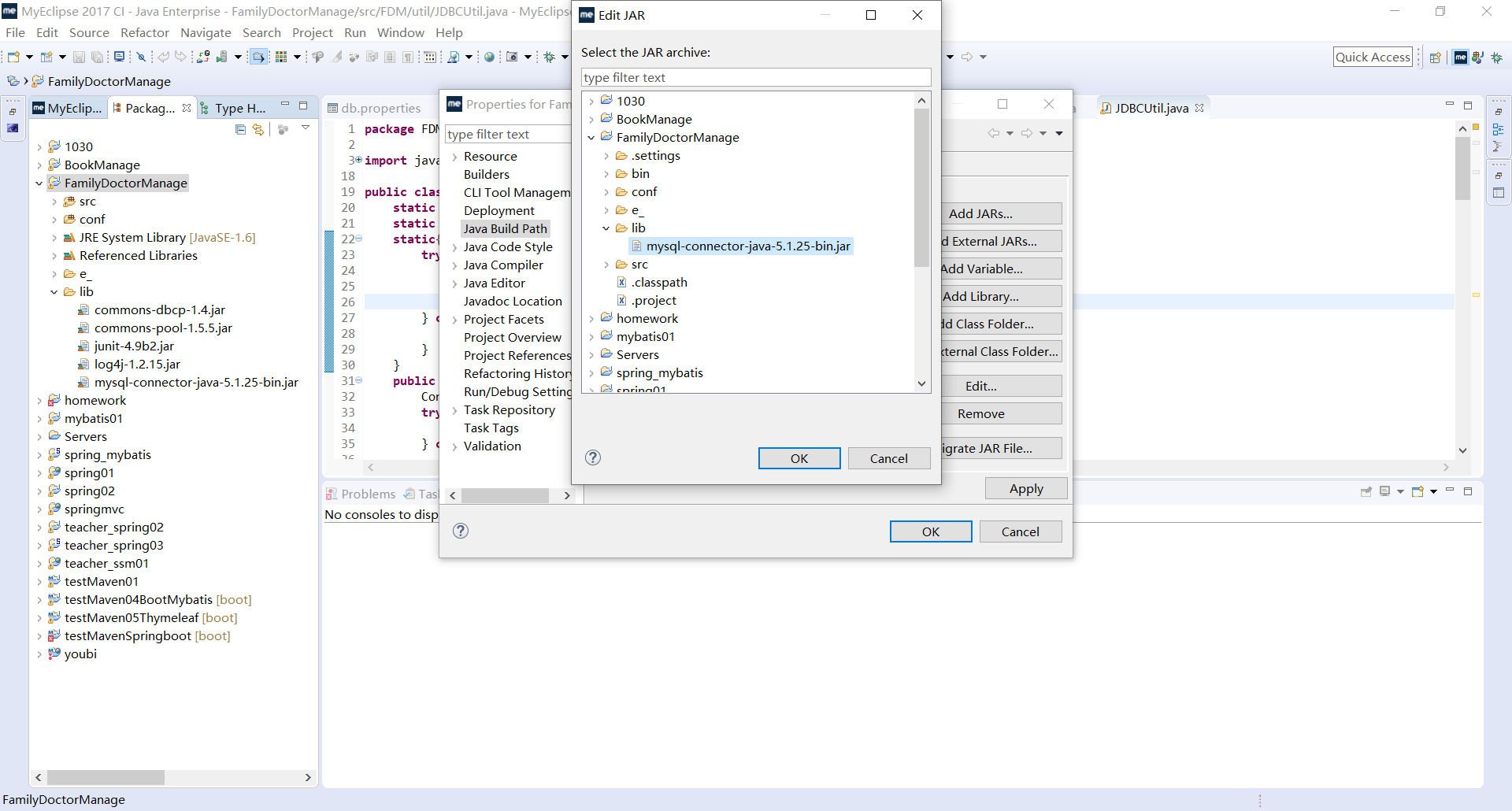
创建完成后就是调用了，调用的方法是

call 存储过程名(传入参数);

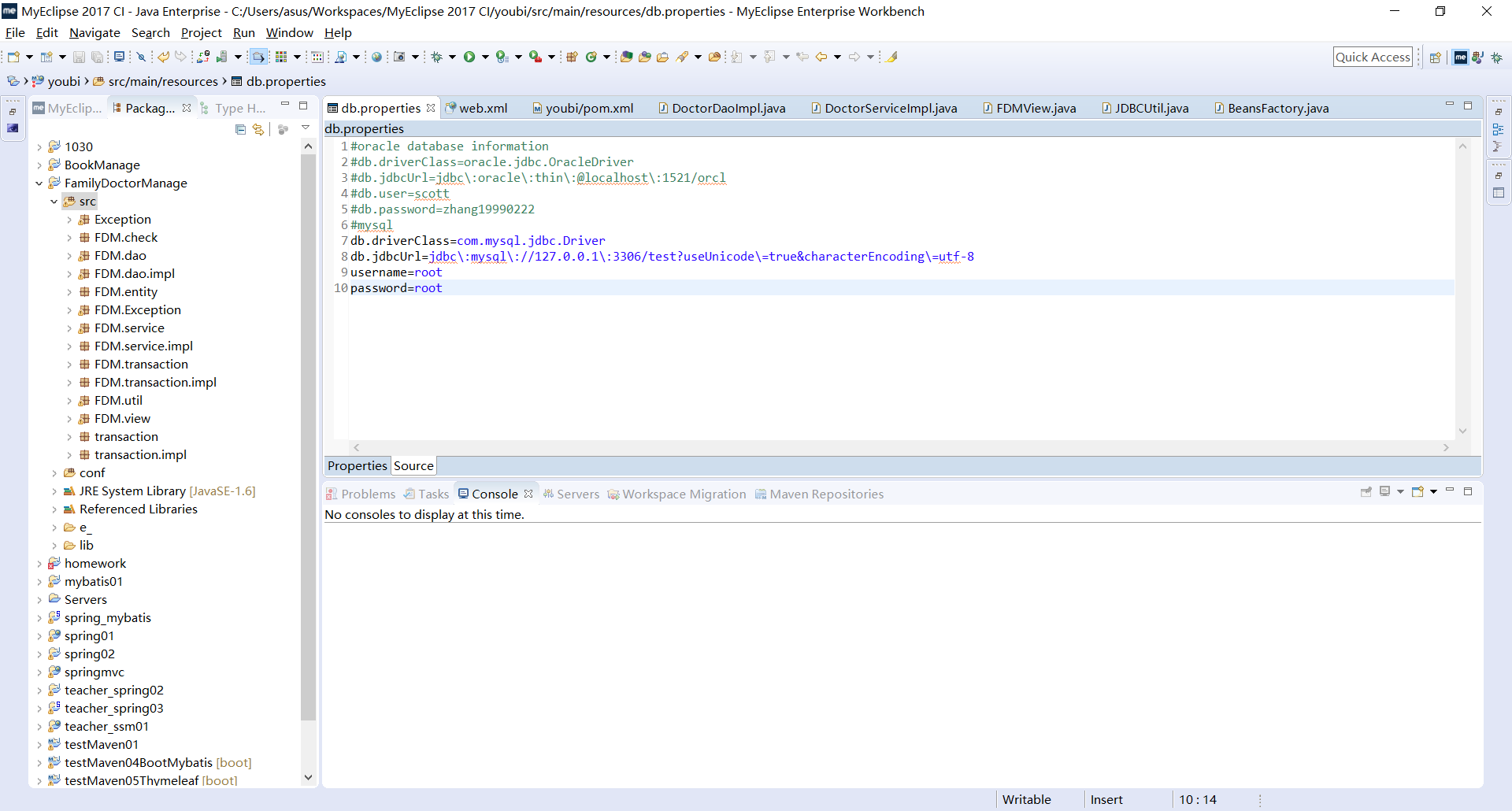
调用之后MySql就会依次执行存储过程中的语句

* 1. 自定义JDBC工具技术

JDBC，全称为Java Database Connectivity，意为java数据库连接，是用于规范java后台程序与数据库连接的一种接口，它是一种规范，而不是一种实现，对于不一样的数据库，其实现方式都有所不同，这里针对MySql数据库进行实现。

要完成JDBC的第一步操作，需要用到一个外部导入的包，即java与数据库的连接包

，将该jar包下载下来后，添加到lib目录中，然后就可以在项目中导入。

与数据库的连接，需要读取配置文件，将连接信息存储在文件中。

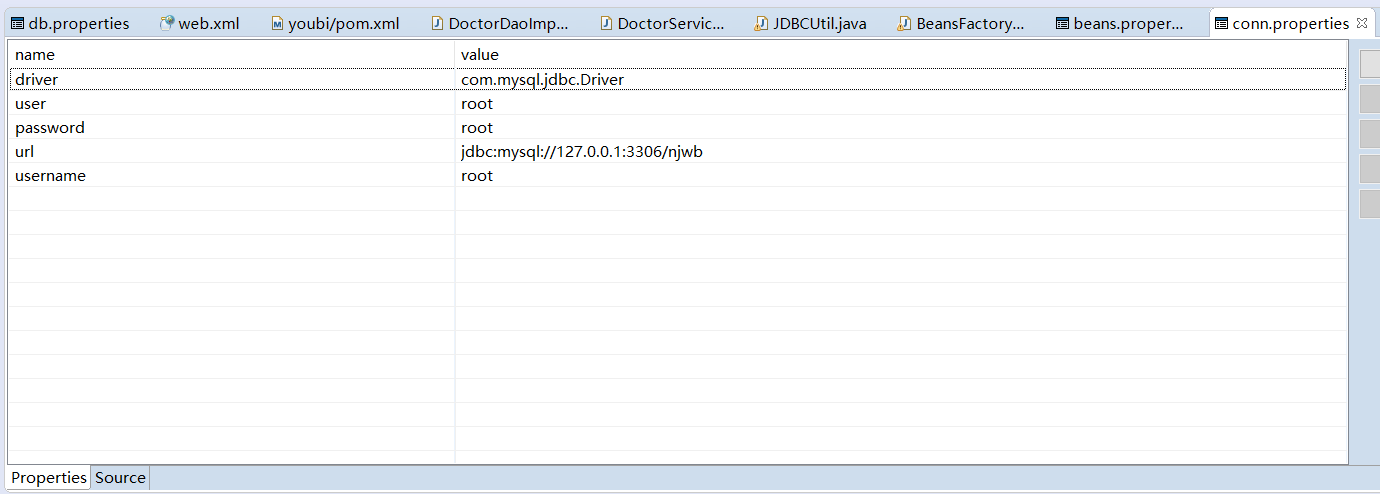
与数据库的连接，将会采用数据库连接池技术来进行。建立连接的数据源数量可以是一个或多个关系型或非关系型数据库，数据库技术的连接池中会将这些可以连接的数据库信息保存，当某一时刻，若我们希望对数据库进行操作，那么就得从中拿出一个或几个连接，从整个系统的数据库连接池中拿出没有处于工作状态的数据库接入，当整个系统对于数据库的所有操作执行完毕之后，此次开启的连接会被关闭并重新收到数据库连接池里面，释放空间。

在过去的操作数据库的方法中，数据库连接需要通过数据库的driver程序来建立，建立之前需要先输入数据库名称和密码，验证成功才行。

这种方法有两个很致命且高能耗的问题：

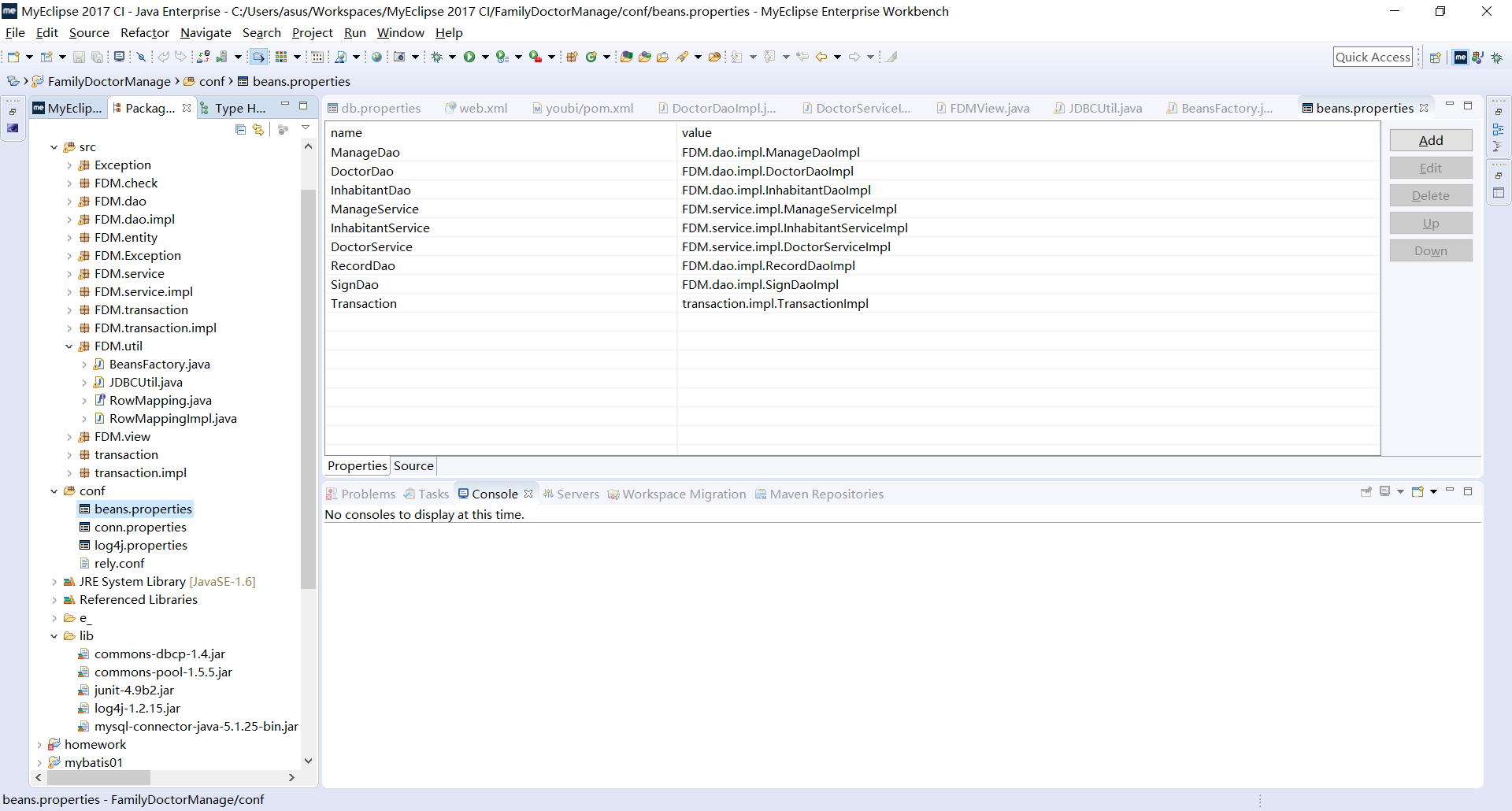
1.与数据库建立连接若过于反复且断开的频次过高，大量的垃圾会留下，大量的资源和时间会因为频繁的连接与关闭被消耗，导致性能降低。

2.需要大量的内存和CPU大量的消耗与使用的有用户名和密码等等都是数据库的连接所要考虑的。

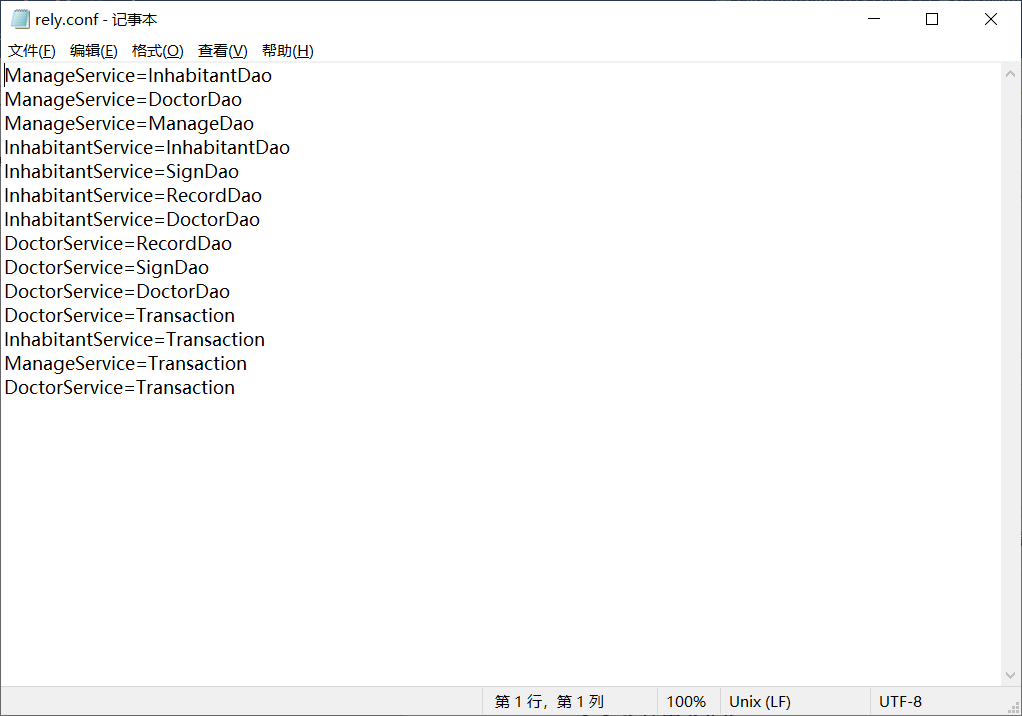
JDBC工具，英文名称JDBCUtil，是一个自定义的java类，其功能包括但不限于通过读取conn.properties连接配置文件来将数据库信息存入文件中方便后续的多次连接、对于数据库增删改查的处理、手动关闭结果集和连接以及连接所创建的statement。配置文件中配置与MySql的连接方法。

* 1. BeanFactory技术

BeanFactory，主要作用是创建对象的工厂，若使用SSM持久层框架，则对象工厂会自动装配实例化对象，此处使用最基本的分层，手动使用对象工厂生成对象以增加对底层代码的了解。对象工厂主要也是通过读取配置文件，对配置文件进行解析，从而生成相应对象。其配置文件内使用键值对形式，类名称与类路径对应。



对象工厂还有一个功能就是读取层与层之间类的依赖，方法名readRely，其原理为解析依赖的配置文件，配置文件的内容是各类之间的调用关系，每个被调用的类对应一个调用的类。

使用对象工厂生成对象，可以将连接和对象的属性都保存，不必生成一个全新的对象然后手动设置属性，大大提高了系统的效率。

* 1. 事务Transaction技术

对于关系型数据库而言，一次操作所涉及的所有数据必须是同步的，比如说从一个数据库表中把数据移动到另一个表中，两个表之间必须才在此次操作之后保持一致，否则数据会出现问题，若一边操作失败，则两张表都必须回到操作之前的状态，此时就需要事务。

事务有4大特性：

原子性：确定事务是最小的，不可以再拆分，事务操作之中要么两边都失效，要么两边都完毕。

一致性(consistency):事务操作的过去与之后，事务执行的双方必须状态相符

隔离性(isolation):不同的事务之间是相互分离的，不会有所干扰

持久性(durability):其产生的后果是永久的，无法撤回的

1. 系统分析
   1. 系统流程概述

此系统管理员账号由终端添加，拥有最高权限，可对居民与医生信息进行增加删除查询以及修改，居民与医生不得自行增加，拥有账号的居民可在系统中查询医生信息，与医生签约，签约后可方进行预约。

* 1. 系统需求分析
     1. 管理员和用户登录

本系统将依据身份信息分为系统管理员登录、医生/居民登陆两种三种登陆。当用户进行登陆操作时，可以自行选择自己的身份进行登录。在登录时，需要用户填入它的登录凭证才可以成功进入系统。登陆凭证包括用户名与密码，这两条通过查询数据库进行判断。只有当用户输入的信息与数据库中的某一条数据匹配时才能进入操作页面，否则将无法成功跳转页面，如当用户名或密码与数据库中查出的信息不一致无法登陆时，页面会弹出窗口：“用户名或密码错误,请重新输入！”。同时在页面上也进行非空验证并设置了提示窗口，方便用户快速解决登陆问题。

* + 1. 用户账户的管理

管理员在登录成功后可以进行医生和居民账户信息的增加、删除与修改功能。居民账户管理与医生账号管理的核心代码大致与相同，故而在此处以医生账号管理介绍。

可以将管理员管理的所有功能显示在同一个页面中。添加医生账号的功能相当于医生注册需要由管理员录入医生信息，进而完成对数据库中的行信息的增添操作，添加完成后，此账户可由医生身份进入本系统完成医生的功能。

医生账号在登录情况下可以对数据库中自己的部分状态进行改动，医生信息的修改不能修改自己的用户名。且自己的姓名不能进行改动。

居民账号修改跟医生账号修改类似。

* + 1. 居民与医生的签约

居民在登录后，可以对所有医生的个人信息进行查看筛选，根据医生id和相应医生进行签约，若已与某医生签约，则无法继续签约功能，提示“您已与医生签约”，需要先进行解约，此外，居民还可以进行解约功能。

* + 1. 居民的预约

居民在与医生签约之后，便可进行预约操作，未签约的居民，若致性预约操作，则会提示，“您还未与医生签约，请先进行签约操作”。若已签约，则可以填写预约信息，进行预约。

1. 系统设计
   1. 系统总体设计

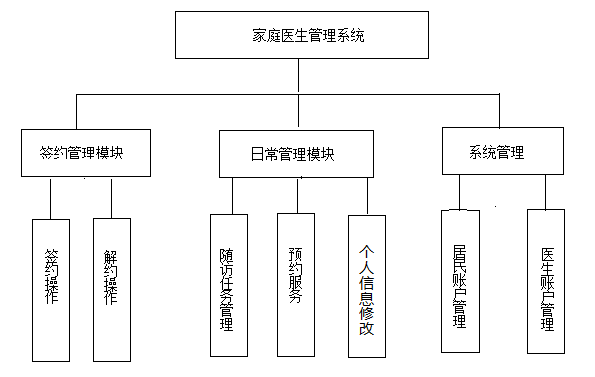
本系统是为了方便家庭医生的工作与管理，在设计开发该系统时，将系统分为居民、医生与后台管理员部分，对于不同身份的用户，他们都可以执行不同的操作内容，逐步实现签约管理、日常管理、预约上门和系统管理四大模块的开发。

* 1. 功能模块设计

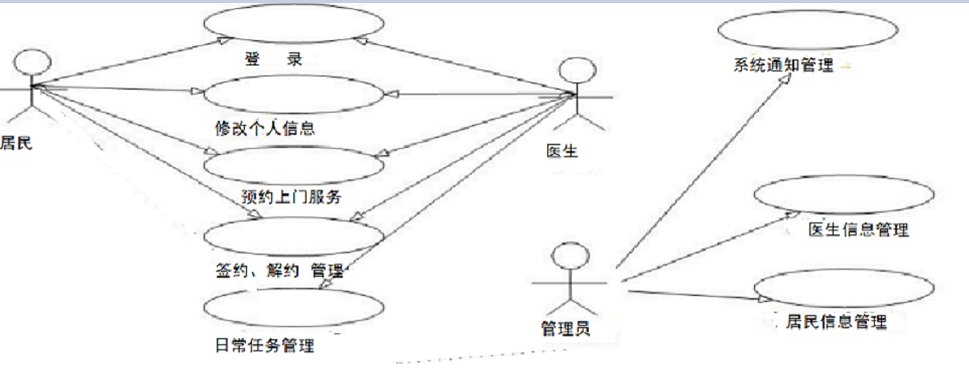
系统功能模块划分表

|  |  |  |
| --- | --- | --- |
| 业务模块 | 功能划分 | 简单描述 |
| 签约管理模块 | 居民对家庭医生的签约 | 居民可以对家庭医生进行查询了解，并选择一位医生进行签约，填写相应的签约信息 |
| 日常管理模块 | 家庭医生随访任务 | 家庭医生在登录后可以对预约检查记录表进行修改，查询指定时间段内的预约记录 |
| 个人信息修改 | 家庭医生与居民在登录后可以对本人的相关信息及密码进行修改操作 |
| 居民预约服务 | 居民可以开启预约，确认上门服务时间、内容等信息。医生在登录后可以进行修改预约信息（同意） |
| 系统管理模块 | 医生信息管理 | 系统管理员登录后可以进行医生账户的添加、删除与修改，同时可以查询所有医生信息，完成对医生信息的管理操作，医生用户的默认密码000000 |
| 居民信息管理 | 系统管理员登录后可以进行居民账户的添加、删  除与修改，同时可以查询所有居民信息，完成对居民信息的管理操作，居民用户的默认密码000000 |

家庭医生管理系统总模块图



家庭医生管理系统总用例图



* 1. 数据库设计

manage管理员

在MySQL数据库中创建系统管理员表，该表只设置一条数据，具有系统最高权限。

该表是用来存放系统管理员信息的，其对应的各个字段名、类型和关系如表4-1所示。

表4-1 manage表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 是否为主键 | 表示 |
| id | int | N | Y | 管理员id（唯一标识），主键自增长 |
| username | varchar(20) | N | N | 管理员姓名 |
| password | varchar(20) | N | N | 管理员密码 |

doctor医生

doctor表是用来存放医生的信息对应的，其各个字段名、类型和关系如表4-2所示。

医生id以Y开头后接4位数字，如Y0001，Y5238等。

表4-2 doctor表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 是否为主键 | 表示 |
| id | int | N | Y | id（唯一标识），序号自增长 |
| doctorid | varchar(20) | N | N | 医生id以Y开头后接4位数字id的表示 |
| doctorname | Varchar(20) | N | N | 姓名 |
| password | Varchar(20) | N | N | 密码 |
| phone | Varchar(18) | Y | N | 手机号 |
| email | Varchar(40) | Y | N | 邮箱 |
| sex | Varchar(10) | Y | N | 性别 |
| medicaltime | date | N | N | 从医时间yyyy-mm-dd |
| place | Varchar(50) | Y | N | 科室（专科） |

inhabitant居民

inhabitant表是用来存放居民信息的，其对应的各个字段名、类型和关系如表4-3所示。

居民id以J开头后接4位数字，如J0001，J4236等。

表4-3  inhabitant表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 是否为主键 | 表示 |
| id | int | N | Y | 居民id（唯一标识），主键自增长 |
| userid | varchar20 | N | N | 以J开头后接4位数字id的表示 |
| username | Varchar(20) | N | N | 姓名 |
| password | Varchar(20) | N | N | 密码 |
| phone | Varchar(18) | Y | N | 手机号 |
| email | Varchar(40) | Y | N | 邮箱 |
| address | Varchar(50) | Y | N | 详细地址 |
| sex | Varchar(10) | Y | N | 性别 |
| brith | date | Y | N | 生日 yyyy-mm-dd |
| state | Varchar(60) | Y | N | 基本情况(血型、既往病史等) |

sign1签约

sign表是用来存储居民和医生的签约关系，一个居民只能跟一个医生签约，一个医生可以服务多位居民。其各个字段名、类型和关系如表4-4所示。

签约单号以QY开头后接4位数字，如QY0001，QY5238等。

已经签约张三医生，并预约明天上午10点检查

是否可以解约再重新签约李四，预约明天上午12检测（在没有预约的情况下可以解约并重新签约）

表4-4 sign表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 是否为主键 | 表示 |
| id | int | N | Y | 签约单号，主键自增长 |
| signId | Varchar(20) | N | N | 以QY开头后接4位数字id的表示 |
| iid | int | N | N | 居民id（外键）唯一 |
| did | int | Y | N | 签约医生id（外键） |
| signTime | date | N | N | 签约时间yyyy-mm-dd |

record预约

record表是用来存储居民健康检查相关信息内容的，其各个字段名、类型和关系如表4-5所示。

预约单号id以YY开头后接预约时间的年月日，后接自增4位数字序号，如YY201710050001，YY201705160125等。

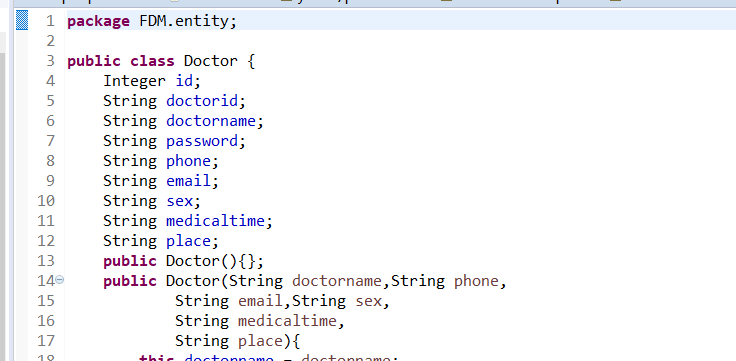
表4-5 record表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 是否为主键 | 表示 |
| id | int | N | Y | 健康档案id（唯一标识），主键自增长 |
| recordId | varchar | Y | N | 以YY开头后接4位数字id的表示 |
| iid | int | N | N | 居民id（外键） |
| appointmentTime | datetime | N | N | 预约上门时间yyyy-MM-dd hh:mm |
| item | varchar | N | N | 检查项目 |
| inspect | varchar | Y | N | 检查结果(只能由医生添加) |
| supply | Varchar | Y | N | 备注信息（医生或居民取消预约） |
| createAt | datetime | Y | N | 预约创建时间yyyy-MM-dd hh:mm:ss |

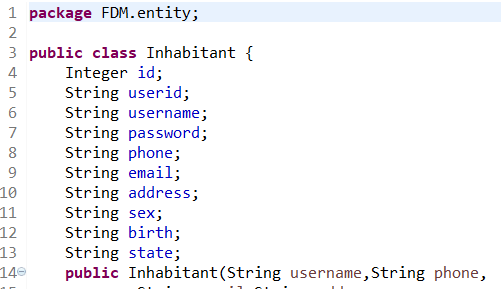
1. 系统实现
   1. 程序设计

要实现一个系统，首先要进行分层，本系统采用最基础的view-service-dao-entity分层，每个表映射一个实体类，放入entity中，一共有五张表，所以entity中有五个类

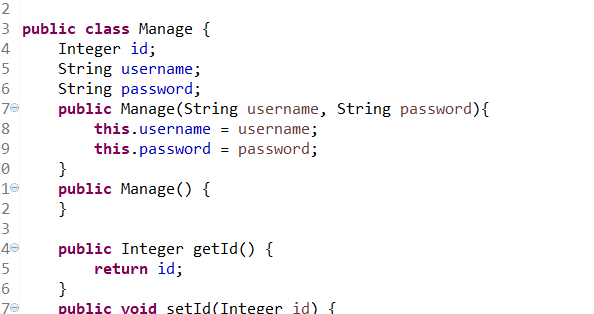
Doctor.java类



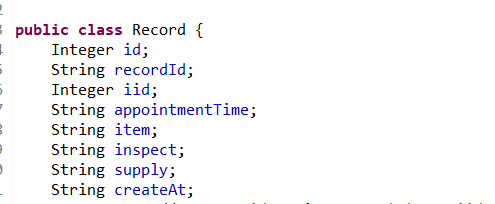
Inhabitant.java类



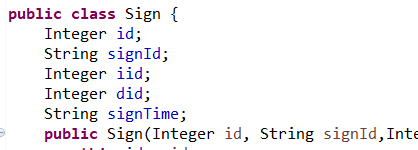
Manage.java类



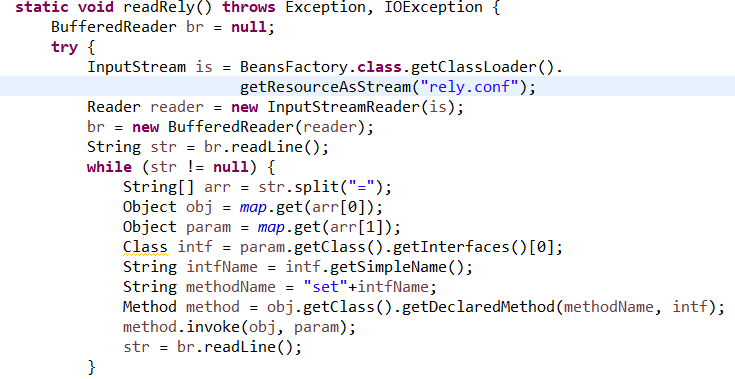
Sign.java类



Record.java类



由于没有用到框架，没有涉及到自动装配和依赖注入，所以在每个实现类中加入相应的构造方法用于生成实体对象，对象的属性会在对象工厂生成对象时读取依赖文件自动调用set方法设置



每个表映射一个dao，写接口和实现类，Dao层主要进行数据库语句的操作，这些操作都放入dao层中，通过调用JDBCUtil，将数据库语句和语句的参数传入JDBCUtil来执行后续的操作。

此外，因为居民和医生编号的前面要加一个标识符，所以需要用到存储过程来添加医生、居民、签约和预约。

create procedure insertD(name varchar(20),ph varchar(18),em varchar(40),s varchar(10),mt date,pl varchar(50)) begin insert into doctor(doctorid,doctorname,phone,email,sex,medicaltime,place) values('a',name,ph,em,s,mt,pl); update doctor set doctorid=concat('Y',lpad((select last\_insert\_id()),4,'0')) where id=(select last\_insert\_id()); end//

create procedure insertI(name varchar(20),ph varchar(18),em varchar(40),ad varchar(50),s varchar(10),bir date,st varchar(60))

begin

insert into inhabitant(userid,username,password,phone,email,address,sex,birth,state) values('a',name,pwd,ph,em,ad,s,bir,st);

update inhabitant set userid=concat('J',lpad((select last\_insert\_id()),4,'0')) where id=(select last\_insert\_id());

end

create procedure insertS(i int,d int,s date)

begin

insert into sign(signId,iid,did,signTime) values('a',i,d,s);

update sign set signIid=concat('QY',lpad((select last\_insert\_id()),4,'0')) where id=(select last\_insert\_id());

end//

create procedure insertR(i int,app datetime,it varchar(20),sup varchar(50),cr datetime)

begin

insert into record(iid,appointmentTime,item,supply,createAt) values(i,app,it,sup,now());

update record set recordId=concat('YY',lpad((select last\_insert\_id()),4,'0')) where id=(select last\_insert\_id());

end//

在Service服务层中，通过对象工厂生成带有属性的相应Service对象，对象工厂同时读取rely.conf配置文件，将dao作为属性给service对象赋值。

对于每种可以登录的对象各写一个service的类用于执行他的功能，故对管理员、居民、医生各生成一个service，将他们能执行的功能各自写在他们自己的service类中。管理员的service中执行医生和居民账户的增加删除修改查询四个功能，由于要管理医生和居民，所以一共需要八个方法。居民的service可以进行查询所有医生信息、预约、签约以及修改部分个人信息的功能。医生的service可执行查看预约签约信息与改动部分个人信息的功能。医生与居民信息的改动不能重新设置用户名。姓名也不能做出改动。

管理员在登录成功后可以进行医生和居民账户信息的数据库基本操作功能。居民账户管理与医生账号管理的核心代码大致与相同，故而在此处以医生账号管理介绍。

可以将管理员管理的所有功能及对应执行的编号放在一起显示。

添加医生账号的功能不可以由用户来完成，需要由管理员录入医生信息，进而完成对数据库中信息的新增，添加完成后，此账户可由医生身份进入本系统完成医生的功能。

在view层中，提供对所有功能的执行，流程的控制在view层中运行，医生居民使用一个分view，管理员使用另一个分view，总体再写一个view用于调用两种用户的view界面执行整个系统的工作流程。首先是登录，选择循环语句，通过switch-case语句选择用户类型，循环中若出现错误则重新循环，而不会报错停止，分别用1~9代表个功能，用户登陆后提示是否要修改密码。

在整个系统分层中还需要加入异常的处理机制，下层捕获并分类抛出异常，上层则接收并处理这些异常，使整个系统能够稳定运行。

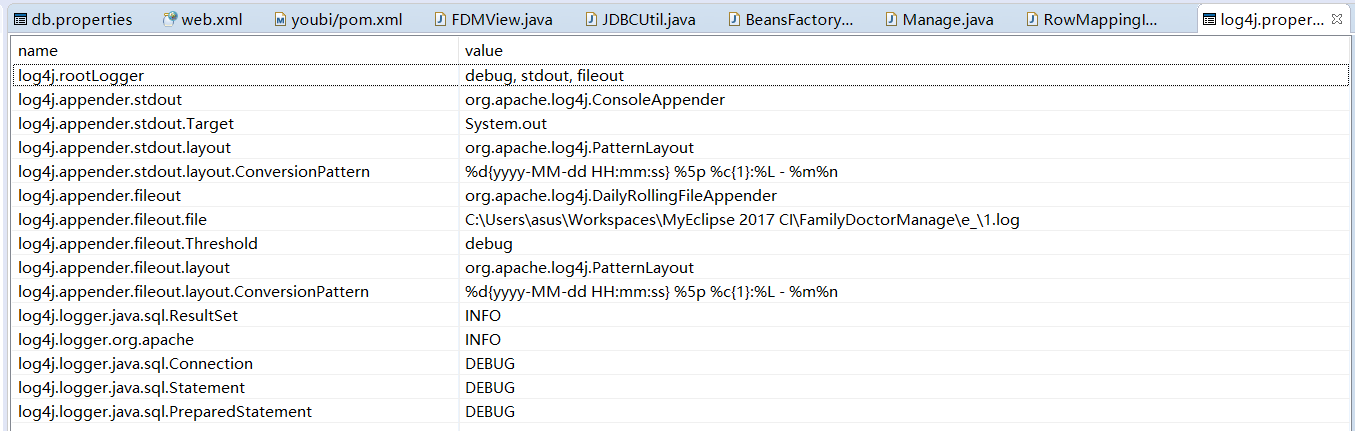
此外，还需要将各异常具体细分，以便分类处理，还需要加入事务的启动、确认交付以及遇到异常时事务的回滚，以防数据库数据出现异常。

事务中需要编写基本的功能，即事务的启动和事务的交付，以及遇到问题时的回滚机制，在开始中需要将提交方式设置为手动提交，在这三个功能之中，每个功能都需要获取当前的连接，从而执行数据库操作。

编写RowMapping行映射类，写在方法内部中，作为内部类，实现数据库查询后数据的实体化功能，将单条数据映射到单个对象中，将多条数据映射到集合List中，用于后续数据的调用显示。

编写垃圾处理机制，当连接之后要执行的所有操作执行完毕后，需要手动将连接关闭，而当行映射的功能使用完毕后也需要将结果集关闭，释放内存空间，使整个系统流畅运行。

为了优化异常的处理机制，使各类异常能够分情况被处理，将医生异常、居民异常和管理员异常分开实现，写在异常包中。

整个系统的各操作结果需要记录到日志中，同样通过配置文件来设置。

log4j.rootLogger为日志根配置语法，主要用于设置要将日志输出的位置及文件名称，debug为日志的优先级，后面跟着的为日志信息的输出位置。

appenderName为log4j提供的appender类，下面给该类中各属性赋值。

Target=System.out：默认情况下是：System.out,指定输出控制台

file的值为接收输出信息的文件，用于写入日志信息。

conversionPattern配置日志输出信息

1. 系统测试
   1. 测试任务

任务主要包括测试系统的以下几个方面

(1)功能性：所实现的软件其起到的作用可以满足基本需求并且符合阿里程序设计规范;

(2)可靠性：给定一定的要求和time，程序是否能够持续执行不崩溃，且效率不会降低多少；

(3)易使用性：对于一个刚上手的系统，要做好相应的注释，为用户提供好的上手体验，让用户能够付出较少努力便能熟练使用系统;

(4)效率：在预定的规则下，运行此系统按流程操作所消耗的资源，包括内存和cpu等，测试其执行效率，避免卡顿;

(5)可维护性：在系统正常流程下，当系统外部发生改变或内部抛出异常时，修正错误所需要做出的努力达到什么样的程度;

软件为主的企业成功基本都有一个必要条件，那就是软件的质量是否过关，软件质量的重要性是无与伦比的。传统文化中的质量管理理念与软件工程的质量并无太大的差别，只是后者主要对软件的许多特征进行了针对性的修改。

* 1. 测试目标

选择软件测试方案主要取决于软件测试的出发点。如果测试软件是为了排查尽可能多的异常，那么测试的只想部分就应该是软件比较繁琐的位置抑或过去出错明显更多的地方。如果测试软件的目的是让用户能够放心的使用自己的系统，给他一颗定心丸，那么此次测试针对的部分就应该是在日常的操作中我们可能会遇到的问题。

对于不同的企业，其测试目的也不尽相同；对于相同的企业，其测试目的也一般不会全然相同，所要执行的测试所要针对的领域和量级也不相同。

首先，测试并不光是为了找出系统中存在的问题。凭借理解问题出现的个中缘由和问题的特性，可以对项目的管理者找出现今使用的系统执行流程中的的过失与问题施以援手，才可以进行后续改善。同时，选对分析的途径也能帮助我们专一于相关渠道给出具体的查错手段，是此次测试具有更高的可靠性。

再者，即使没有发现问题，测试也是有它自己的独特作用的，评定结果的手段之一即是一次贯穿始终的测试。这一点可以通过事无巨细而周密的有所依托的增长模型来应验。

* 1. 测试方案

1. 启动管理系统，观察有无运行错误，能否正常显示第一步操作提示；
2. 选择登录身份，首先输入1，选择管理员，输入账号密码看是否能登录；
3. 依次执行1~8运行对于医生和居民们的数据库各种基本操作，观察有居民和没有居民时显示信息是否正常，选择已设置功能之外的数字，提示输入无效，请重新输入；
4. 退出功能，结束系统流程；
5. 再次启动管理系统，首先输入选择用户而非管理员，接着选择医生，输入doctorid和密码进行验证操作，进入系统提示是否需要修改密码，可选择否跳过，依次执行医生的各项操作权限；
6. 退出到用户类型选择界面，选择居民，继续通过输入个人凭证登录系统，流程与医生相同；
7. 依次输入1、修改个人信息2、查询所有医生信息3、签约4、解除签约5、预约6、解除预约7、退出
8. 若执行过程中不会报错，也不会自动退出，执行成功后数据库数据也一致，则表明测试成功。
9. 总结

经过本次毕业设计的实践，我对一个管理系统的设计与建立有了初步的了解，懂得了开发一个项目的基本流程，如何搭建架构和给系统进行简单的分层，如何进行问题的分析和解决，和一些空指针异常的排查，锻炼了我的动脑能力，提高了我寻找和排除问题的能力。经过这一次实战，提高了我编写与编辑代码的熟练度与效率，通过编译环境所报出的错误，一层一层寻找到错误的源头，对于我理解整个系统提供了莫大的帮助。

参考文献

《Thinking in Java》Fourth Edition - Bruce Eckel President, MindView, Inc.647~672

《Java编程语言在计算机软件开发中的应用》 李燕；吴海珍; 常章梅；

《浅谈Java中Mysql数据库的连接与操作》 张振超; 吴杰; 陈序蓬；

《基于Java和MySQL的图书馆信息化管理系统设计》 黄文娟；

《Electronic Laboratory Notebook Assisting Reflectance Spectrometry in Legal Medicine》 Lioudmila Belenkaia, Michael Bohnert, Andreas W. Liehr

致谢

首先要感谢我的论文指导老师、人工智能学院的马杰良老师。马老师给我们提出了具体的方向，有问必答，我们的问题都能得到及时的解答，也对我的论文提出了指导性意见，在平时的学习中也是悉心教学，对我们之后的选择起到了很多积极的影响。

此外，还要感谢各学科的老师，是他们教会了我分析问题解决问题的方法，传授给我这些基础的知识，使我不会无从下手。

最后，我还要感谢读到这里的每一位老师和同学，以及我的家人，没有你们就没有今天的我。