**《面向服务的软件系统》实验报告**

**——实验一：服务构件/服务系统的设计与实现**

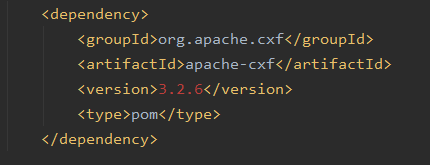
姓名： 孙源 学号： 1173710226

姓名： 原浩宸 学号： 1172910113

姓名： 董希尧 学号： 1171000106

* 实验1 基于Apache CXF框架的服务开发环境搭建

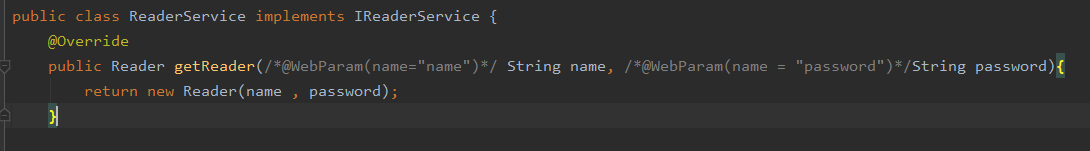
1. 项目中的pom.xml中的dependencies是什么？有什么作用？

pom.xml里面的dependencies是用来描述本项目的依赖，例如本项目需要apache.cxf 那么我们就在这里加入关于apache.cxf的依赖，这样就可以

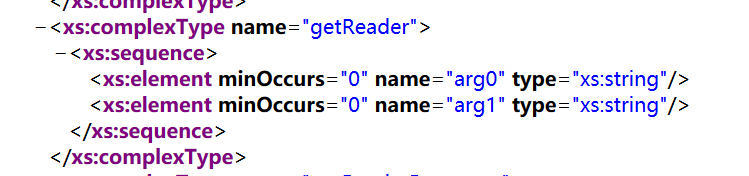
不用再去下载了。只需要在dependencies中写好之后，maven就会帮你自己下载所有依赖到本地，方便项目的迁移和依赖关系的更新和升级。

1. 服务接口中方法参数中@webparam注解用与不用的实验结果是否会有变化？

@WebParam是表示在网页的方法的参数，以下图函数getReader为例

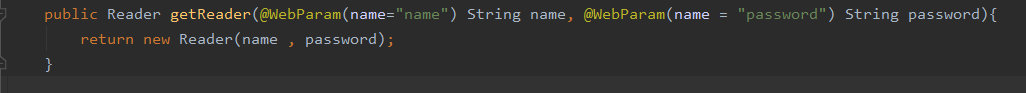


我们先把@WebParam注释去掉，然后启动服务器，打开我们的wsdl服务界面。



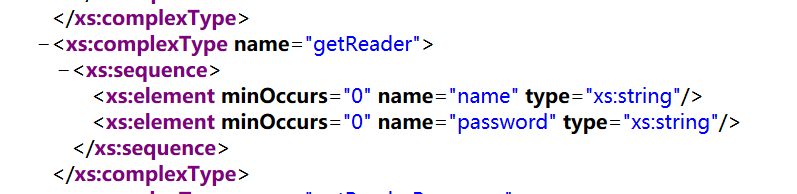
发现getReader方法的两个参数变成了arg0和arg1，等于说对外服务的变量名字无法让使用服务的用户知晓。

我们再把@WebParam加回去，



重启服务器

再次打开服务界面，



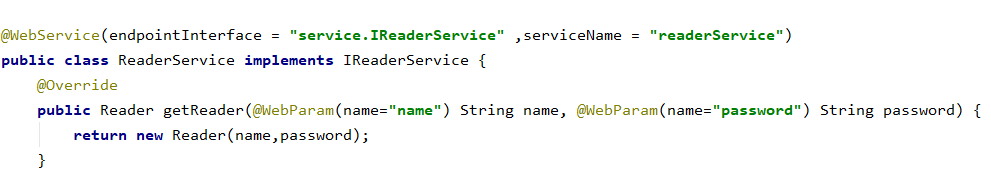
这个对外服务的字符串就有了变量名（name和password）。

1. 方法实现中@webservice注解中的参数endpointInterface和serviceName分别代表了什么？

endpointInterface代表服务接口的路径

serverName代表了服务对外发布的名称

以ReaderService为例



service.IReaderService是ReaderService的接口，同时也是服务请求和服务实现的对接接口

readerService是发布的服务名称

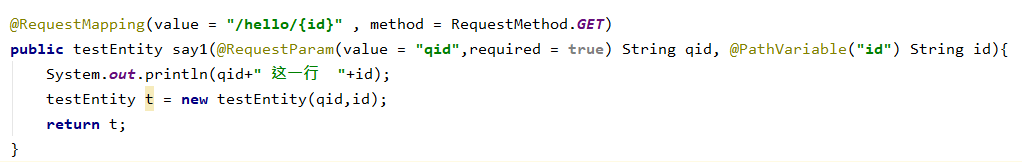
* 实验2 基于SpringBoot框架的服务开发环境搭建

1. 注解@RequestParam和@PathVariable分别表示什么？

@RequestParam表示必需传递某个参数

@PathVariable表示将url中的占位符作为参数

以HelloController里的say1方法为例：

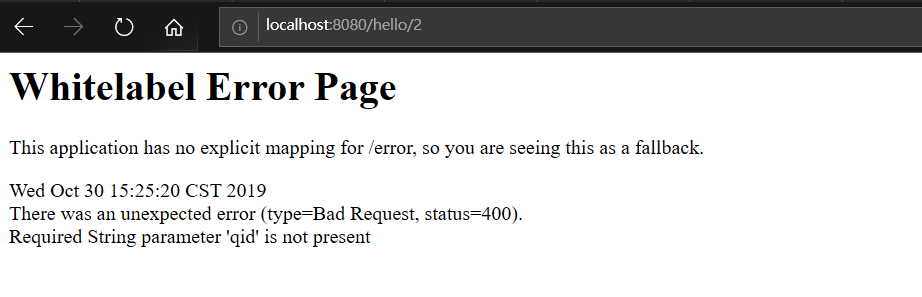


@RequestParam（value = ”qid”,required = “true”）表示必须传递“qid”参数，required=true表示必需，若required=false则表示不必需，此时和没有该注解效果一致

@PathVariable（“id”）表示在@RequestMapping注解中 取url的占位符中名为id的参数的值，作为参数传递到该方法中。

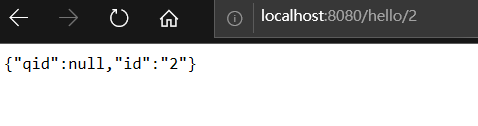
1. 用以下URL：<http://localhost:8080/hallo/2>访问本实验的服务会得到什么结果？如何解释该结果？

结果如图所示：



因为正如1所说，两个注解总共需要url两个占位符来传递参数，然而url中只有一个{id}，导致无法获得所必需的“qid”参数，故出现了错误。

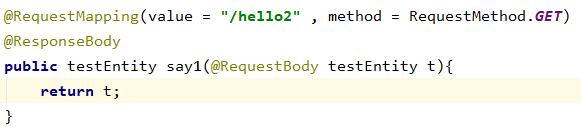
只需要将@RequestParam中的required属性改成false（或者直接将该注解删除），“qid”参数就不是必需的了，修改之后再运行，可得到：



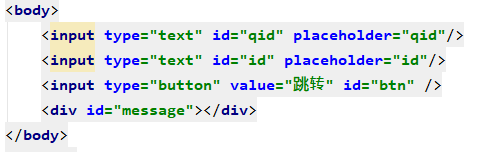
1. 用上课讲过的JQuery的代码编写一个可以正确访问服务中say1方法的客户端程序。

采用ajax将数据以json格式进行传递

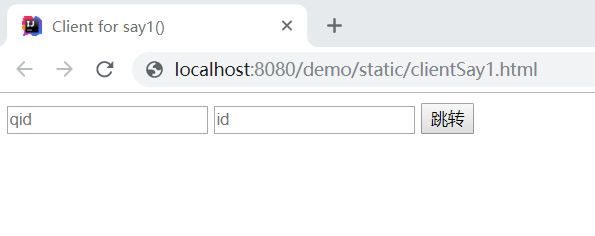
将服务方法say1做一点修改



Html页面



在相应的html中输入信息，即可传递到后台



* 实验3 服务构件/服务系统开发

请在实验报告里写清楚以下内容：

1. 需求分析

房源信息管理服务：房源信息的增删改查。

其中房源信息具体包括，房屋的地理位置location、面积area、价格price、所处楼层floor、建筑时间buildTime、产权年限timeLimit、所属用户user、户型type、状态state（目前是否销售）。

涉及到的服务需求有 1.添加房源并上传房屋信息 2.依据房屋信息查询满足客户条件的房源 3.修改房屋状态，可在待销售与不销售之间转换

评价服务：针对指定房源进行评价，评价可查看。

其中评价涉及到用户和房源，用户指发出评价的用户，房源指被评价的房源。可通过输入房屋来查看所有该房屋的评价，也可以通过由指定客户对指定房屋做出评价。

用户中心服务：用户可注册，编辑个人信息，用户信息可查看。

用户可以通过输入姓名进行注册，并可以修改和查看用户信息

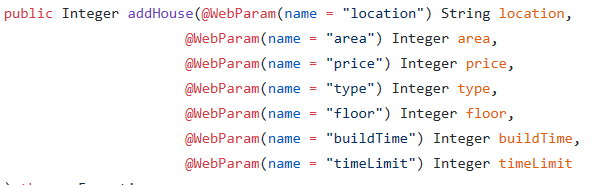
1. 服务接口设计

房源信息管理服务：

根据功能可分为增、改、查

增加主要有：

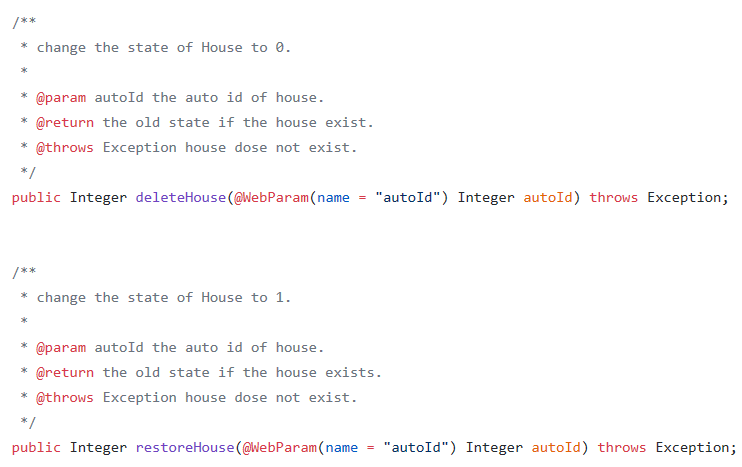
AddHouse添加房源



修改主要包括：

DeleteHouse 将房屋状态改为不可销售

RestoreHouse 将房屋状态改为可销售



查询主要包括

searchByLocation 依据地理位置查询

searchByType 依据房屋户型查询

searchByArea 依据房屋面积范围查询

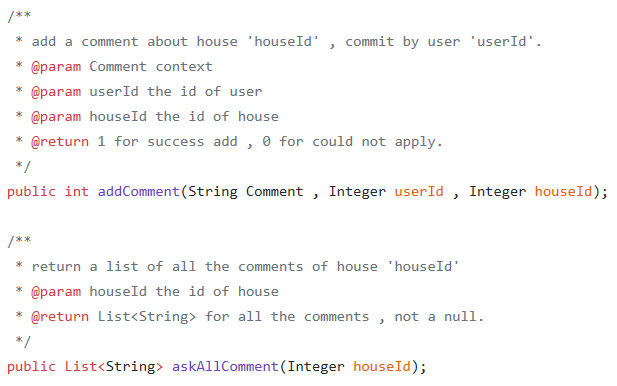


评价服务：

主要有两个服务接口

addComment实现由指定用户向指定房源进行评价

askAllComment实现查看指定房源的所有评价



用户中心服务：

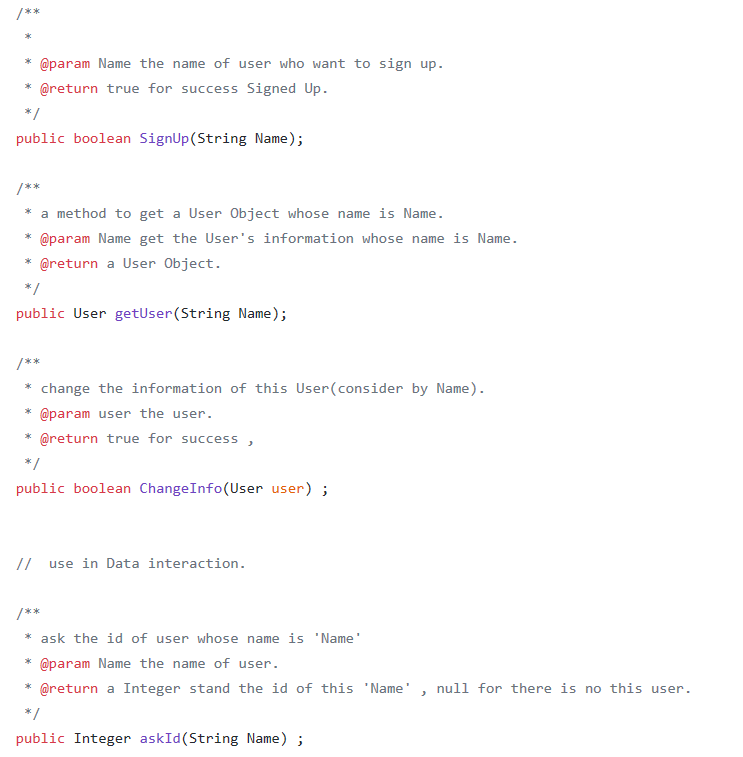
主要有四个服务接口：

SignUp：实现用户注册，输入用户姓名即可

GetUser：查询用户，依据用户姓名进行查询

ChangeInfo：修改用户信息

AskId：用于数据库查询以及后台传递

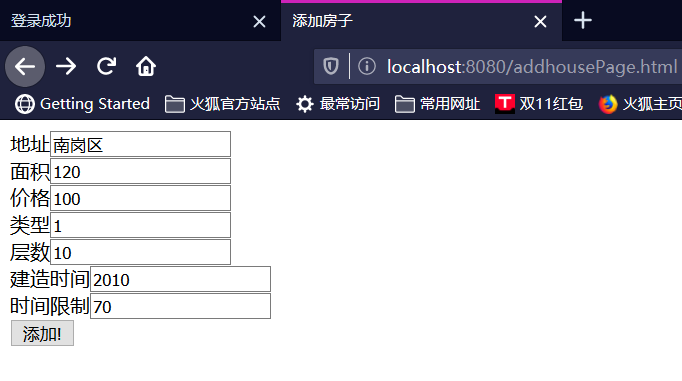


1. 服务接口使用方法与约束

**房源信息管理：**

1. 增加房源信息，在浏览器输入

[http://localhost:8080/addhousePage](http://localhost:8080/hallo).html即可进入增加房源页面,而后信息通过[http://localhost:8080/H](http://localhost:8080/hallo)ouseOp/addHouse服务传递到后台，并返回添加成功信息



1. 查询，以通过房屋地理位置查询为例，在浏览器输入

[http://localhost:8080/House](http://localhost:8080/hallo)Info/location/hit

会返回地址为hit的所有房源的信息

其他查询类似，在/location的位置修改为对应的信息名称即可，比如/area , /price等

3.修改，以将房屋修改为不可销售状态为例，通过服务[http://localhost:8080/H](http://localhost:8080/hallo)ouseOp/deleteHouse，获取需要修改的房屋id，进行修改

**评价服务：**

1.添加评价，通过服务[http://localhost:8080/C](http://localhost:8080/hallo)ommentOp/addcomment

实现评价服务，传递评价信息到后台并返回结果信息：添加成功或添加失败

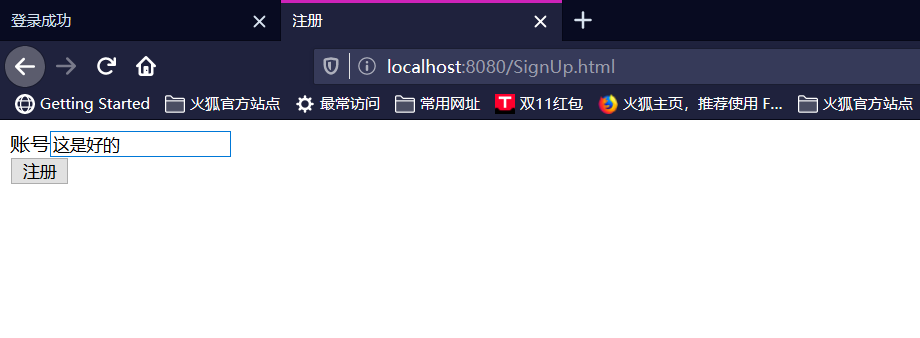
2查询房屋评价，通过服务

[http://localhost:8080/C](http://localhost:8080/hallo)ommentInfo/askhouse/2

查询房屋id为2的房屋评价

**用户中心服务**：

1.注册服务，在浏览器输入 [http://localhost:8080/S](http://localhost:8080/hallo)ignUp.html即可进入注册页面



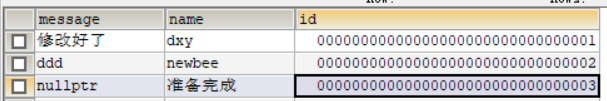
2.修改信息，在浏览器输入[http://localhost:8080/ChangeInfo](http://localhost:8080/hallo).html即可进入账户信息修改页面，在输入框中输入相应信息后，信息就会被传递到服务 localhost:8080/UserOp/changeInfo中

3.查询用户信息 在浏览器输入[http://localhost:8080/U](http://localhost:8080/hallo)serInfo/dxy 即可查询到用户名为dxy的用户相关信息

1. 数据库的设计

我们用了三个表user\_table , house\_table , comment\_table，分别用来存储用户信息，二手房信息，评论信息

* 1. user\_table



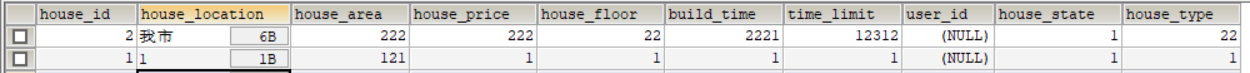
user\_table一共有三个关键字，

message用来存储有关这个用户的信息；

name是表示这个用户的用户名；

id是表示这个用户加入顺序的自增的id（每个用户独一份）。

* 1. house\_table



house\_table一共有10个关键字

house\_id是一个自增id，表示这个这个二手房记录到数据库的id。

house\_location是一个text，用来记录这个二手房的地理位置

house\_area是一个整数，表示这个二手房面积。

house\_area是一个整数，表示这个二手房的价格（元）

house\_floor是一个整数，表示这个二手房楼层数

build\_time是一个整数，表示这个二手房的建筑时间年份

time\_limit是一个整数，表示这个二手房的使用年限

user\_id是一个整数，表示这个二手房的属于者是谁。

house\_state是一个整数，表示这个二手房是待售还是不待售

house\_type是一个整数，表示这个二手房的户型

* 1. comment\_table



comment\_table一共有4个关键字

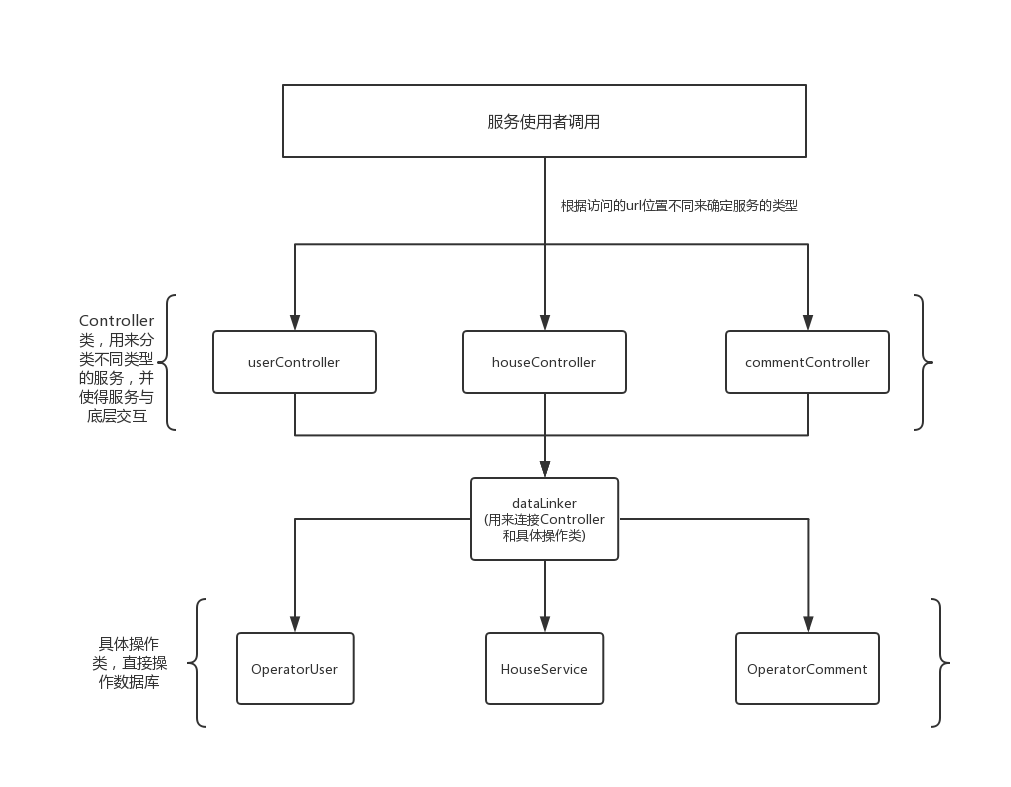
comment\_id是一个自增的整数，表示这个评论的编号

comment是一个text，表示评论内容

user\_id表示评论这个房子的用户的id

house\_id评论目标房子的id

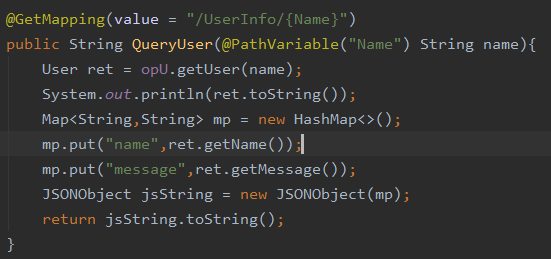
1. 代码实现流程图



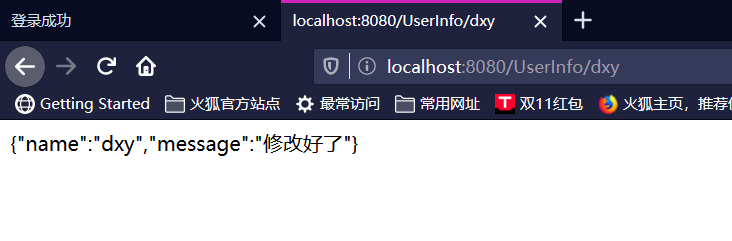
1. 服务接口测试结果

**对于user相关的查询和修改**

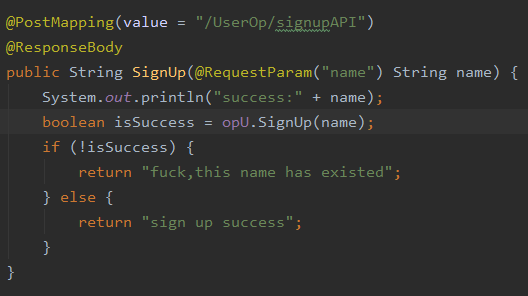
1.查询用户名为{Name}的用户信息



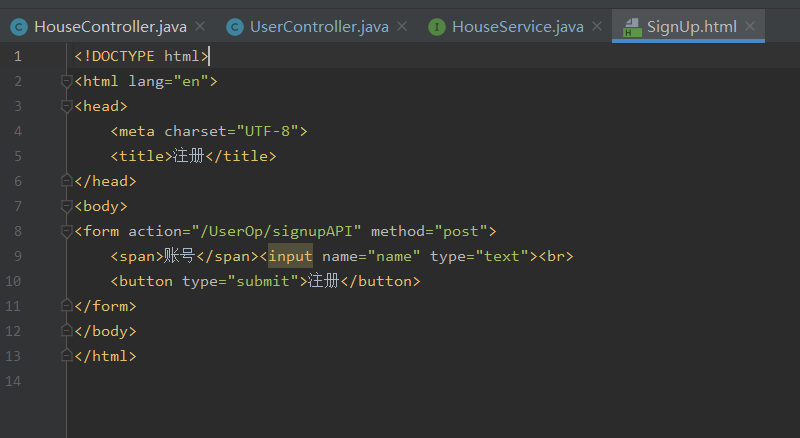
访问<http://localhost:8080/UserInfo/dxy> 可以看见正常显示的查询结果



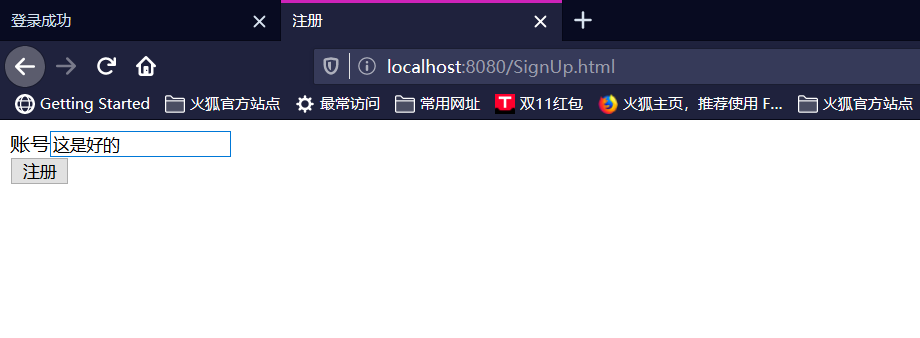
2. 用户注册



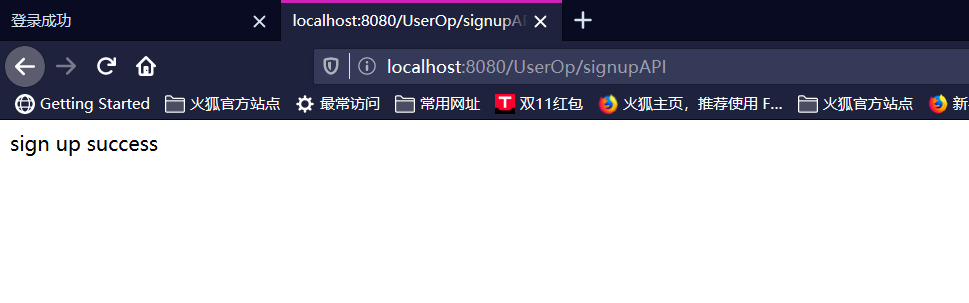
由于使用的post方法，所以我写了一个静态的html页面用于注册



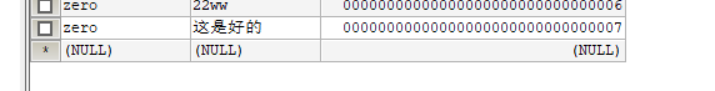
测试



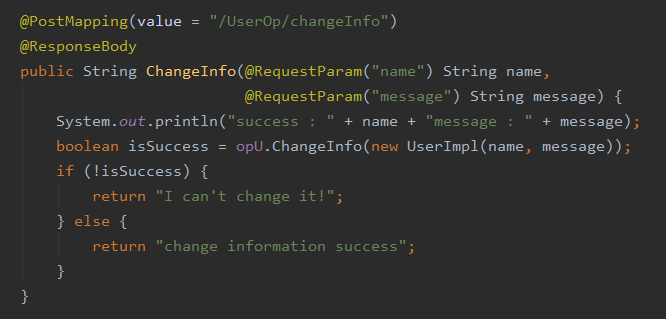
调转到下图界面，注册成功



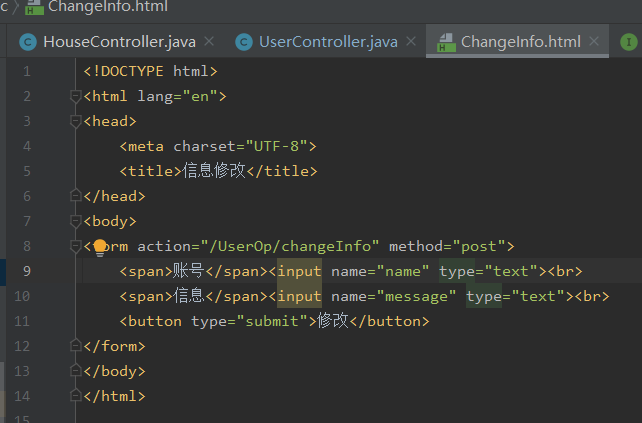
查看数据库，发现信息已经传入。



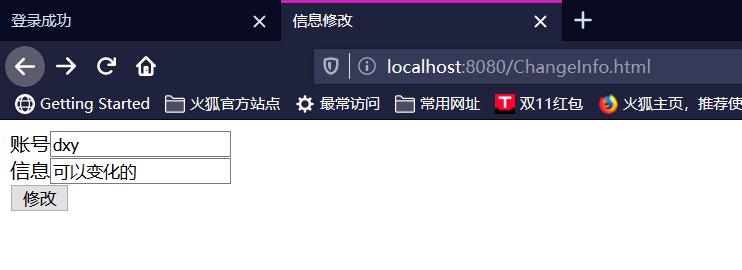
3.修改用户信息

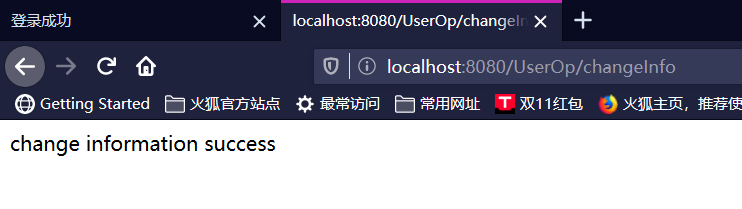


由于是post方法我们也写了一个静态页面

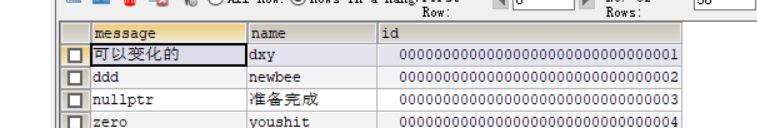


输入信息，按下修改之后显示修改成功





查看数据库，发现已经被修改

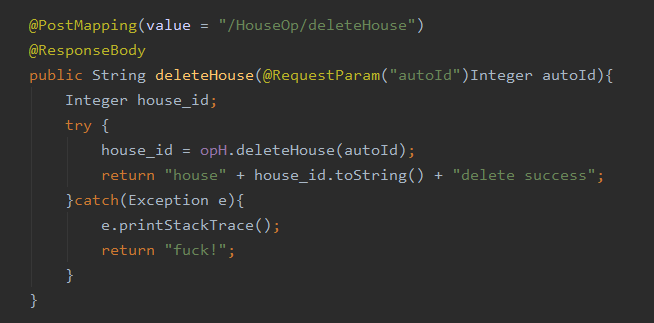


**对于house相关信息的查询和修改**

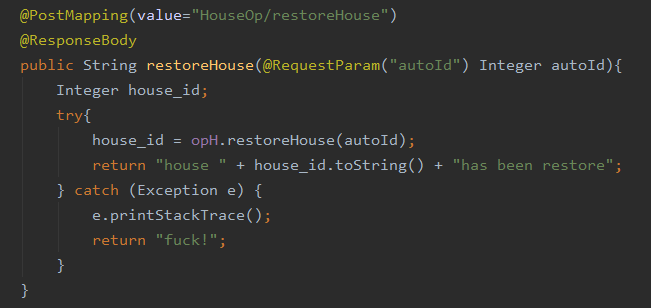
1. 添加/待售/非待售 二手房的信息



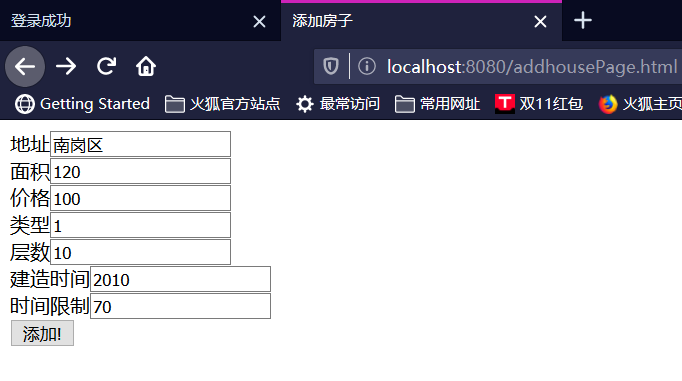
让一个二手房的信息转为非待售

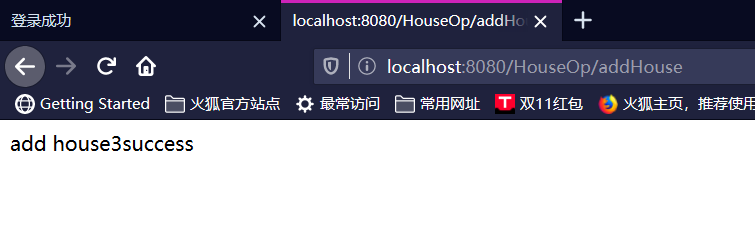


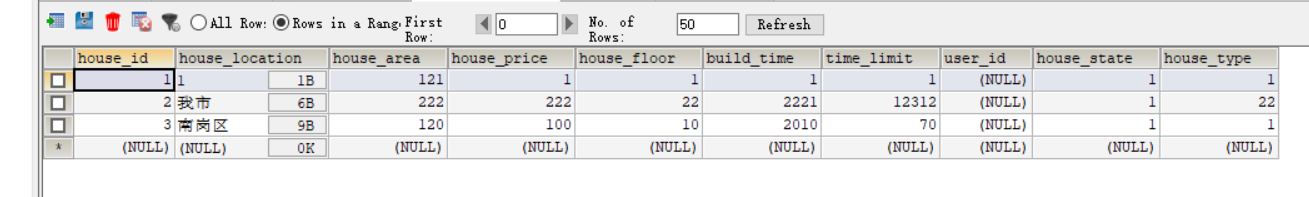
让一个非待售的二手房设为待售的



跟测试signup一样，我也写了一个静态页面来检查这个服务接口







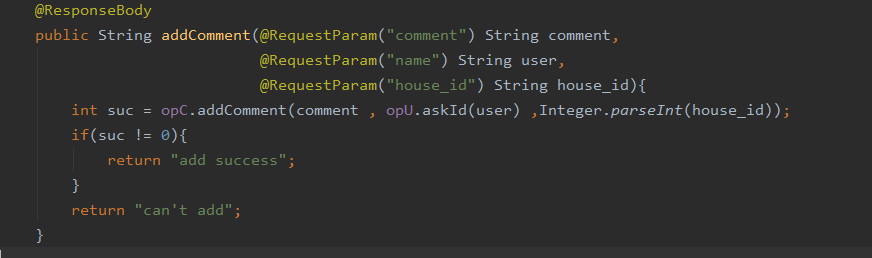
其他需要post的controller测试方法同理。

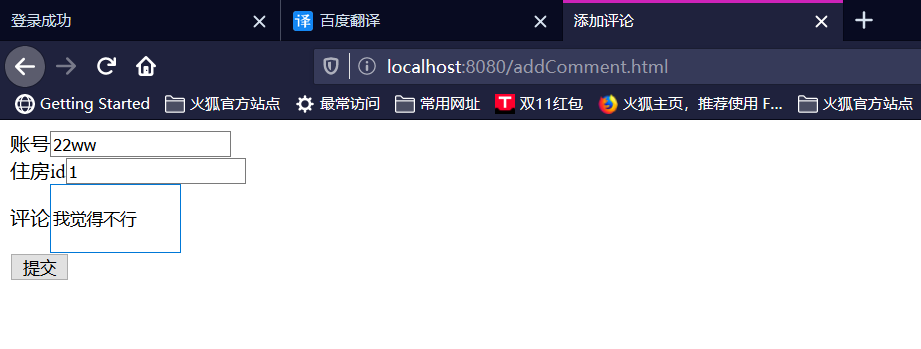
2.对于查询信息的方法

跟上面查询user信息的方法类似。

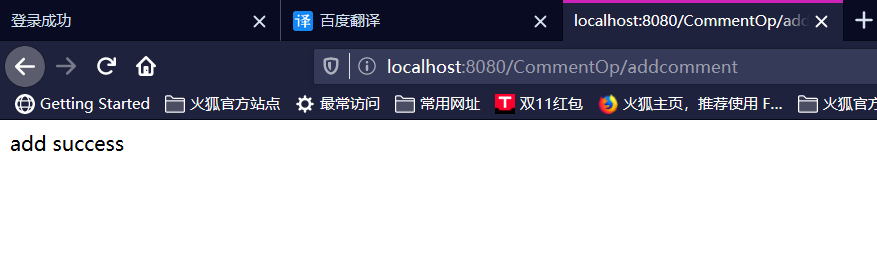
**对于评论查询和添加**

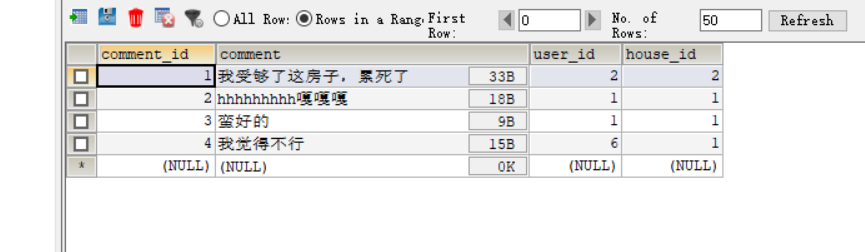
1. 评论添加





用户可以通过这个上图页面进行添加评论的操作（下图为添加成功）

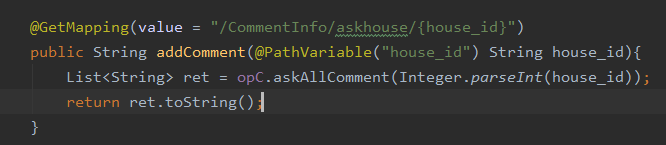




数据库中也都有显示

1. 查询评论

根据住房id查询



（因为是list）下图为查询结果（3个评论完整[逗号隔开]）

