به نام خدا

سیستم های توزیع شده

گزارش تکلیف اول - پیاده سازی سازی RMI

زهرا اخلاقی-۴۰۱۱۳۱۰۶۴

پیش فرض ها:

- چند کلاینت میتوانند به طور همزمان به سرور متصل شوند.
- کلاینت ها از آیپی و پورت سرور اطلاعی ندارند.
- برای اجرای برنامه ابتدا سرور و سپس کلاینت ها اجرا میشوند.
- کلاس سرور و کلاینت در دو برنامه جداگانه اجرا میشوند اما در هر دو RMIRequest برای بازسازی اطلاعات ارسال شده و ارسال اطلاعات وجود دارد.
- اینترفیس های ایجاد شده برای پیاده سازی متد ها از Remote ارث بری میکنند.
- سرور در ابتدا پورتی ندارد و با اتصال به LocateRegistery به صورت داینامیک پورت به آن اختصاص میابد.
- کلاس ها و ورژن های متفاوت روی پورت های متفاوتی اجرا میشوند.

شرح پیاده‌سازی:

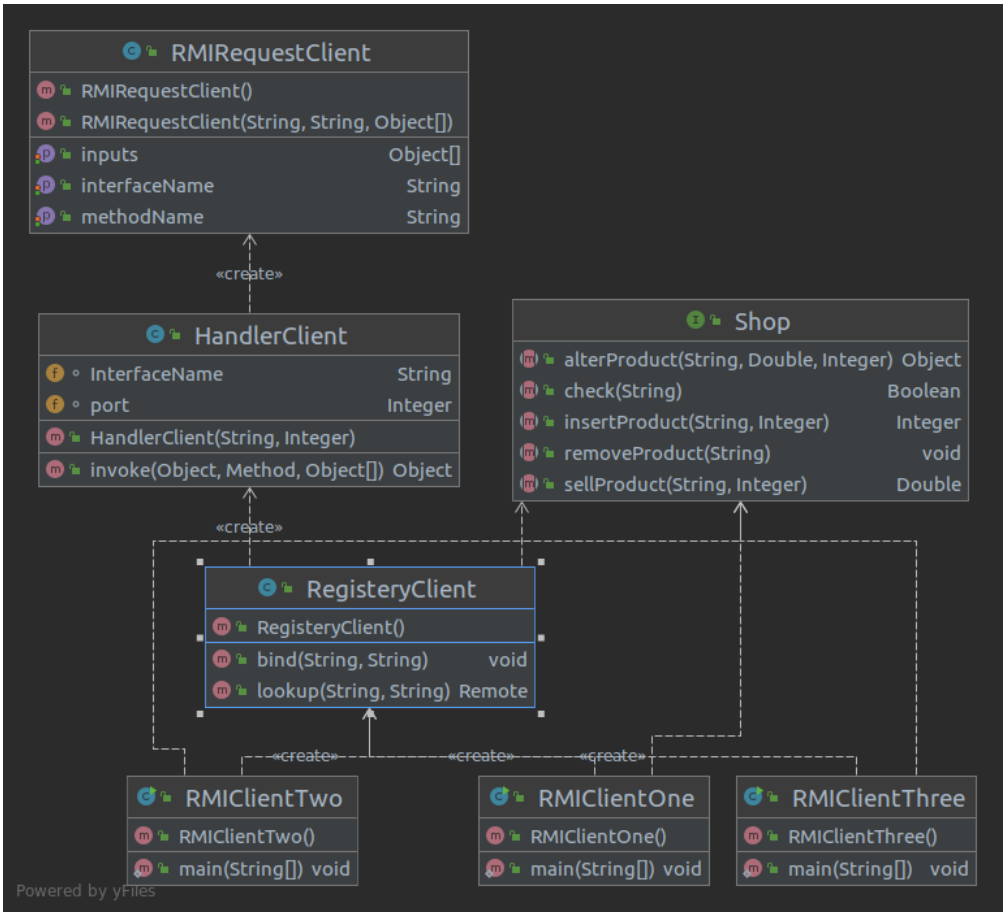
در این اپلیکیشن ، سیستم بسیار ساده ای برای فروشگاه به زبان جاوا پیاده سازی شده، که در آن ویژگی های name, value, amount از کالا ذخیره میشود و برای کاربر امکان افزودن، حذف، خریدن، چک کردن وجود کالا و افزایش تعداد موجودی کالا وجود دارد.

برای ارتباط میان کلاینت و سرور از سوکت استفاده میشود و متد های زیر برای فراخوانی از دور در این فروشگاه در نظر گرفته شده است.

```java
public interface Shop extends Remote {

    public Double sellProduct(String name,Integer amount) throws RemoteException;

    public Boolean check(String name) throws RemoteException;

    public Integer insertProduct(String name,Integer amount) throws RemoteException;

    public Product alterProduct(String newName, Double newValue, Integer newAmount) throws RemoteException;

    public void removeProduct(String productName) throws RemoteException;
}
```

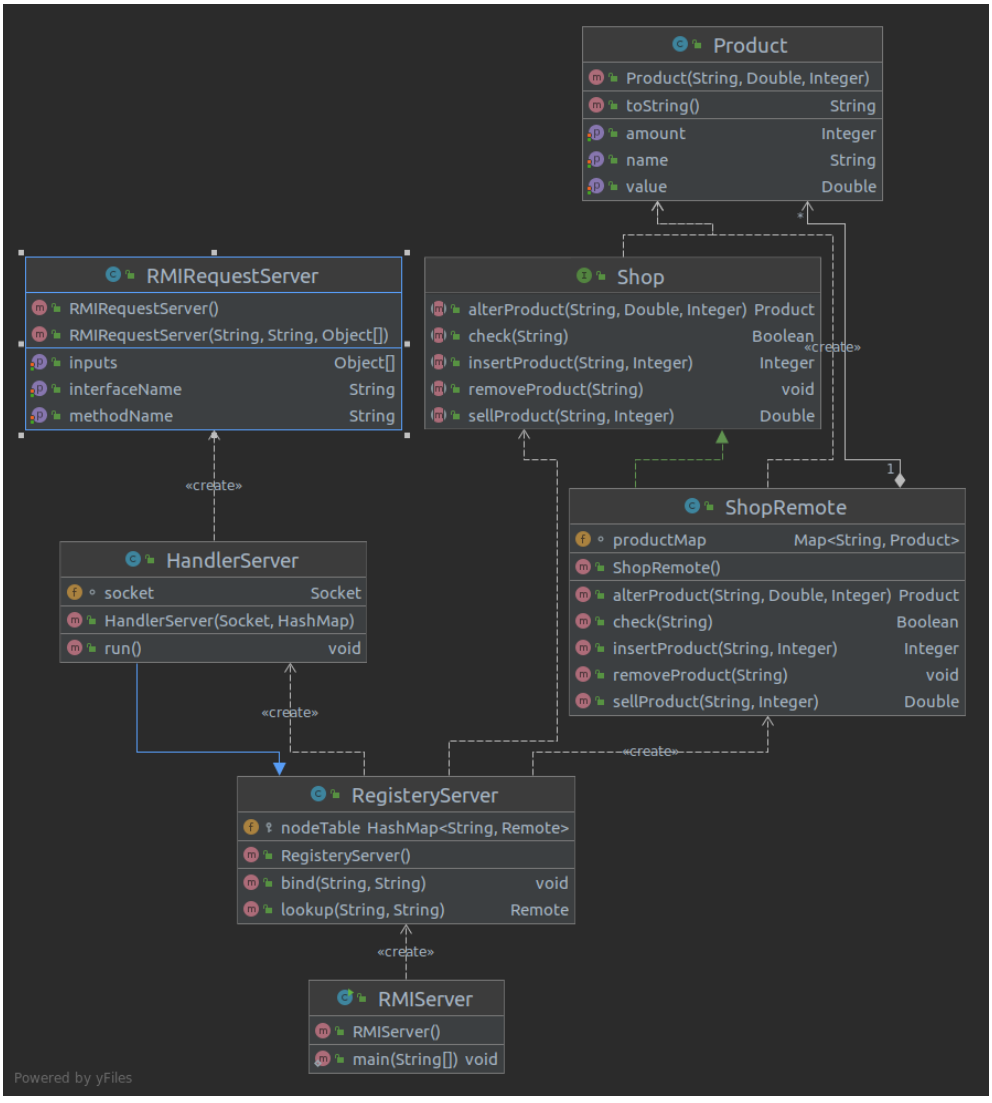
برای پیاده سازی از سه پکیج استفاده شده (Client, Server, ServerRegistery) پکیج ServerRegistery برای یافتن و اختصاص دادن پورت، کاربرد دارد و کلاینت و سرور برای پیدا کردن پورت مناسب برای ارسال اطلاعات به یکدیگر از این پکیج استفاده میکنند. پیاده سازی کلاینت و سرور در پکیج کلاینت و سرور انجام میشود، این دو پکیج از طریق سوکت با یکدیگر ارتباط دارند.

◄ پکیج Client



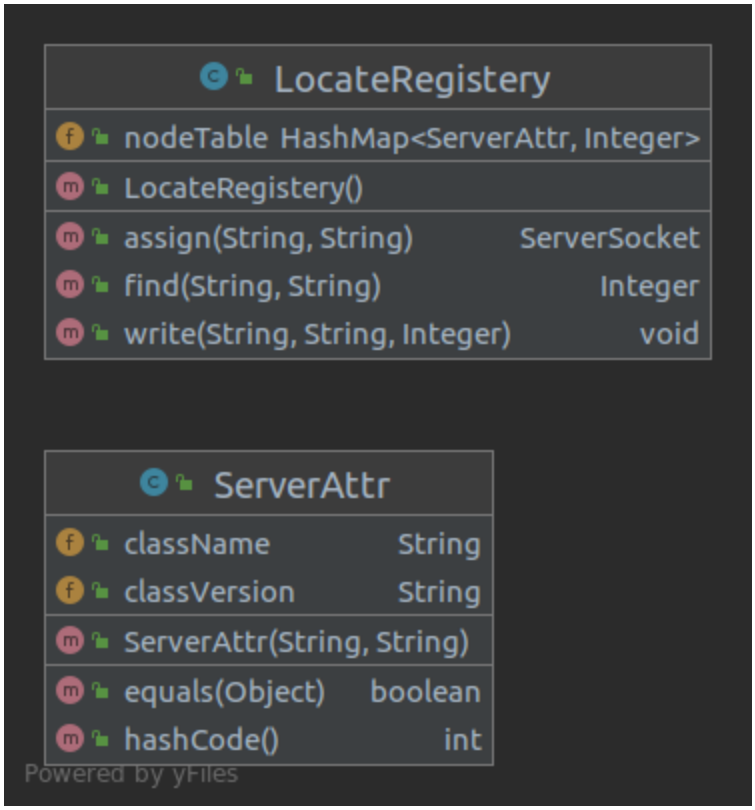
- کلاس های RMIClientOne, RMIClientTwo,RMIClientThree برای تست سیستم استفاده شده اند.
- کلاس RMIRequestClient برای رد و بدل اطلاعات با سرور کاربرد دارد(فرمت درخواست ها).
- کلاس RegisteryClient در ارتباط با پکیج RegisteryServer پورت مناسب را برای ارتباط با سرور براساس className, classVersion میابد.
- کلاس HandelerClient برای ارسال درخواست به سرور و دریافت پاسخ کاربرد دارد.

◄ پکیج سرور



- کلاس RMIRequestServer برای رد و بدل اطلاعات با کلاینت کاربرد دارد(فرمت درخواست ها).
- کلاس RegisteryServer ابتدا پورت مناسب را از LocateRegisteryدر یافت میکند و همواره در حال گوش دادن به آن پورت میباشد و به هر کلاینت که متصل میشود thread مناسب اختصاص میابد.
- کلاس HandelerServer برای ارتباط سرور با کلاینت کاربرد دارد و از اینترفیس Runable را Implement میکند.

◄ پکیج serverRegistery



- کلاس LocateRegistery برای اختصاص دادن پورت مناسب به سرور میباشد و کلاینت به وسیله آن از آدرس سرور را اطلاع میدهد. این اطلاعات در ابتدای اجرا از فایل خوانده شده و در Map ذخیره سازی میشود.
- کلاس ServerAttrحاوی اطلاعات سرور میباشد.

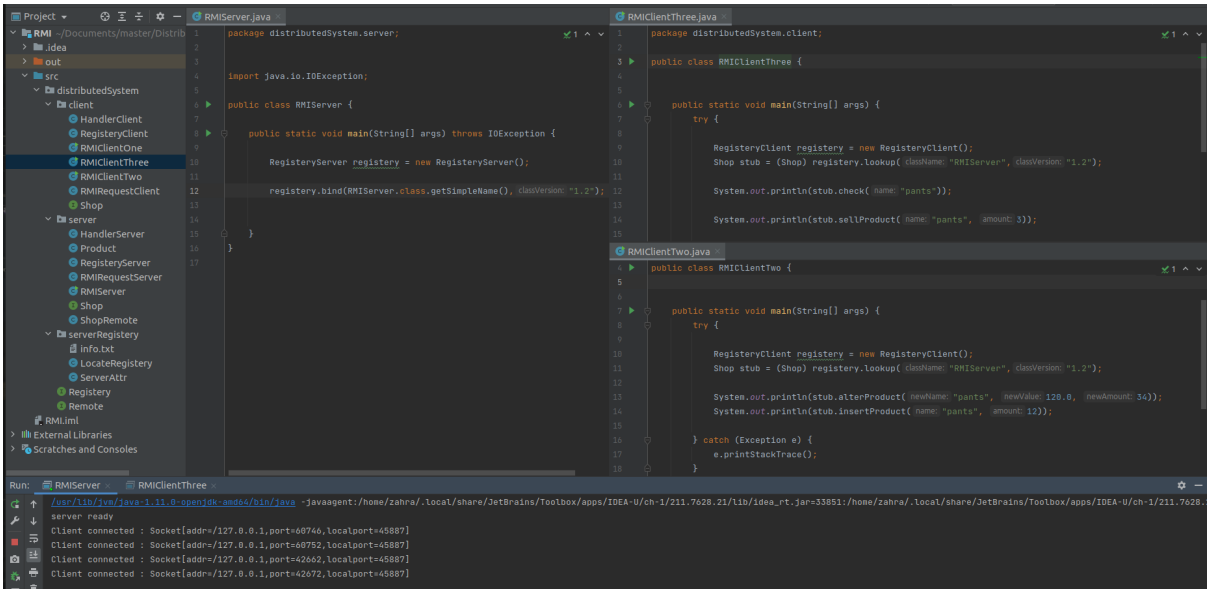
شرح جزئیات:

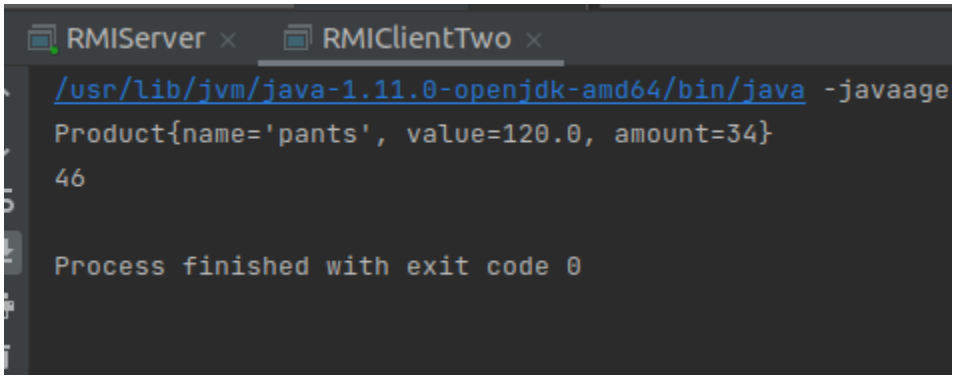
- برای اجرای برنامه ابتدا سرور اجرا میشود (کلاس RMI Server)و سرور همواره در حال گوش دادن به پورت اختصاص یافته میباشد.
- در ابتدا براساس className, classVersion به سرور پورت مناسبی به صورت داینامیک ازمیان پورت های آزاد(در کلاس LocateDiagram در پکیج serverRegistery) اختصاص می یابد.
- برای اختصاص پورت آزاد به سرور از پورت صفر استفاده شده زیرا انجام این کار در زبان جاوا باعث میشود سوکت انتخاب پورت را از میان پورت های آزاد انجام دهد.
- برای ارتباط کلاینت و سرور از سوکت استفاده شده است و آیپی و پورت به صورت dynamic اختصاص می یابد.
- برای ارسال نام اینترفیس، متد و پارامتر های ورودی و مقادیر بازگشتی، کلاسی به نام RMIRequest در سمت کلاینت و سرور تعریف شده است که درخواست ها از طریق آن بسته بندی و ارسال میشوند.
- کلاینت برای ارتباط با سرور از طریق className, classVersion سروری که با آن میخواهد ارتباط برقرار کند، پورتی که سرور روی آن قرار دارد را یافته و با آن ارتباط برقرار میکند.
- اختصاص دادن پورت به سرور و گرفتن پورت توسط کلاینت در پکیج serverRegistery انجام میشود و اطلاعات (className, classVersion و پورت) در فایلی ذخیره میشود و در ابتدای اجرا اطلاعات از فایل خوانده شده و در Map ذخیره میشود.
- سرور بعد از دریافت درخواست از هر کلاینت thread جداگانه ای برای پاسخگویی به آن کلاینت ایجاد میکند، بدین منظور کلاس HandelerServerاینترفیس Runable را Implement میکند.

- در Thread ایجاد شده، پارامتر های موجود در RMIRequest شناسایی شده و فراخوانی براساس آن پارامتر ها انجام شده و سپس مقدار بازگشتی به فرمت RMIRequest بسته بندی شده و به کلاینت ارسال میشود.
- در سمت کلاینت برای دریافت نام و پارامتر های ورودی متد از پروکسی های جاوا استفاده شده.
- یک اینترفیس به نام Remote وجود دارد که هر اینترفیس برای پیاده سازی متد ها در سمت کلاینت و سرور باید از آن ارث بری کند.
- کلاها، تعداد و قیمت آنها در هر بار اجرا شدن معنا دارند و همان مقادیر اولیه را میگیرند(در یک HashMap سمت سرور نگهداری میشود)

اجرا کد:

در این اجرای نمونه ابتدا سرور و سپس کلاینت دوم و سوم اجرا شده اند.





نقاط قوت:

- سیستم قابلیت این را دارد که چند کلاینت به طور همزمان از آن استفاده کنند.
- انواع داده شامل int, char, float, String, Long, short, double, byte میتوانند توسط این سیستم به عنوان پارامتر ورودی و خروجی استفاده شوند.
- کلاینت بدون پیاده سازی و آگاهی از بدنه متد ها میتواند از آنها استفاده کند و آنها را فراخوانی کند (مشابه اینکه متد ها به صورت محلی پیاده سازی شده اند).
- سرور برای پاسخگویی به هر کلاینت thread جداگانه میسارد، بنابراین امکان موازی سازی وجود دارد.
- آیپی و پورت اختصاص یافته به صورت dynamic میباشد از dynamic binding استفاده شده است.
- سرور میتواند برای ارسال اطلاعات به کلاینت از آبجکت استفاده کند.

نقاط ضعف:

- آبجکت های کاربر نمیتوانند به عنوان ورودی متد ها استفاده شوند.

- سیستم برای تعداد زیادی کلاینت تست نشده و امکان مشکلات احتمالی وجود دارد.
- این سیستم پیاده سازی ساده ای از RMI جاوا میباشد و امکانات پیشرفته این کتابخانه را ندارد.
- برای ذخیره سازی کالا ها از دیتابیس استفاده نشده است.