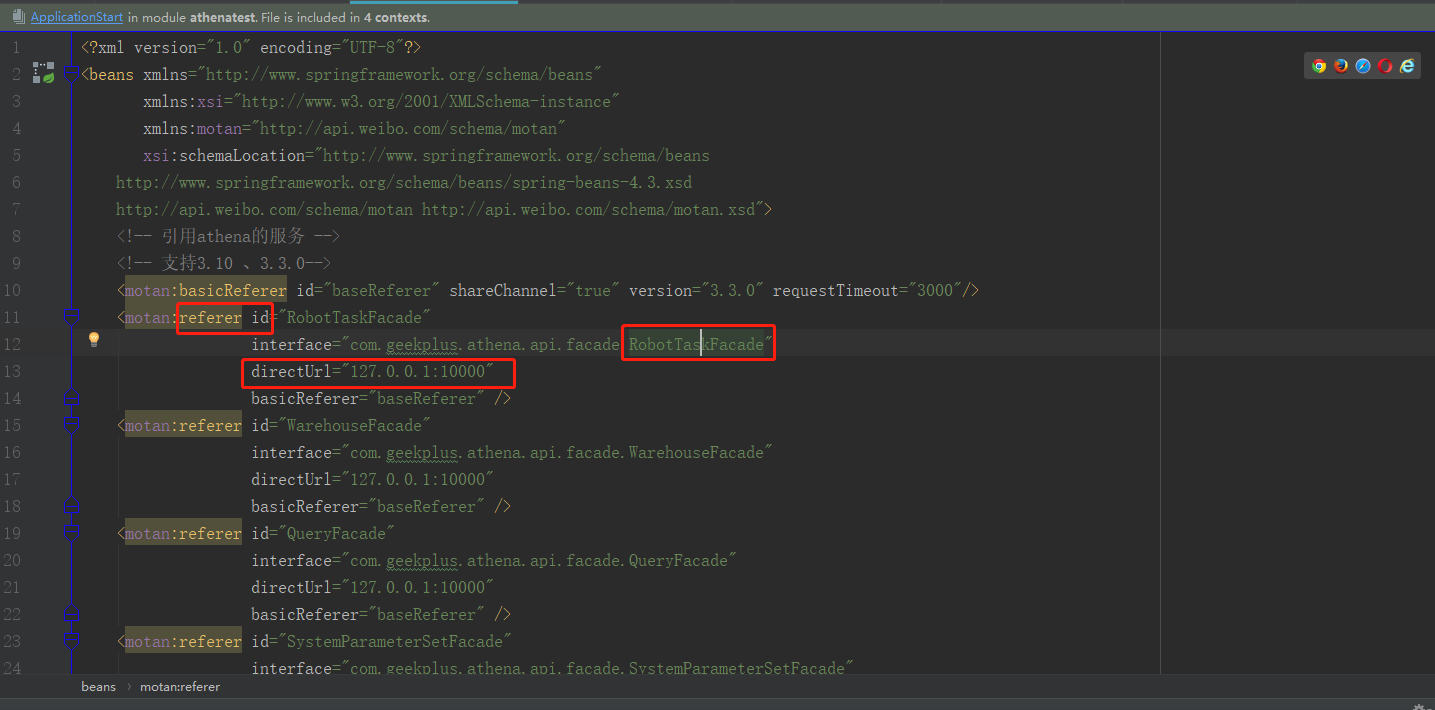
Rms 系统的RPC

1定义客户端（自己是服务端（接口实现方），对方是客户端（调用方））

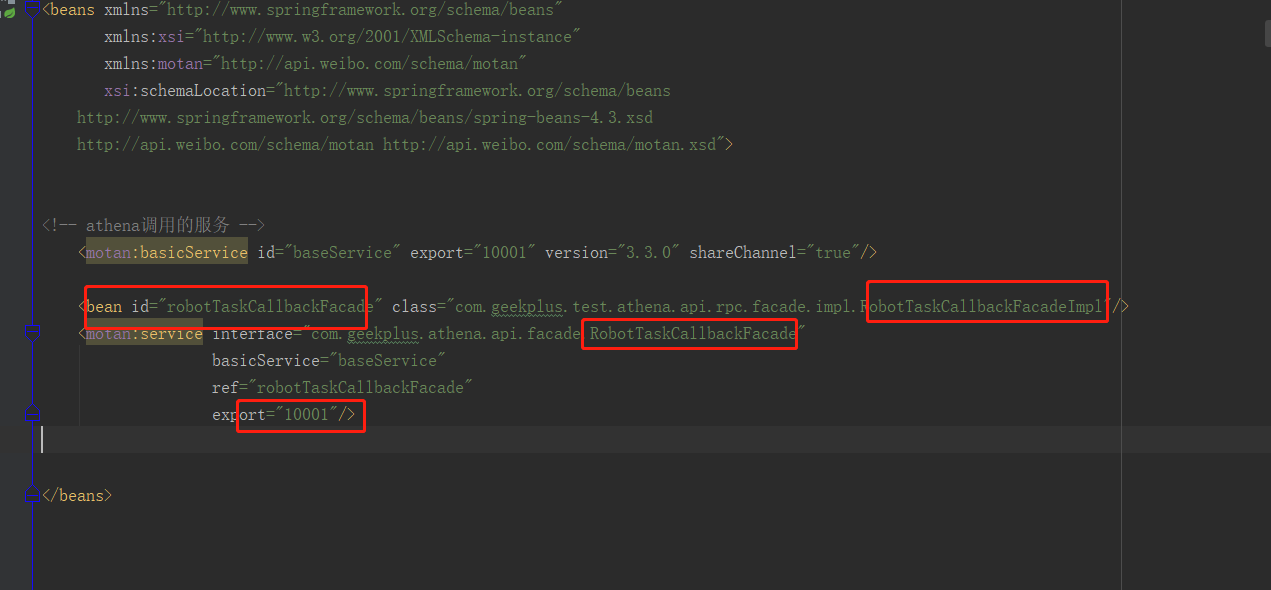


1referer id="RobotTaskFacade"：代表bean 得id名字，

2调用对方的接口是RobotTaskFacade，rms服务端有相应的实现接口，

3.调用的服务ip地址是127.0.0.1，对方的ip服务机上10000端口实现RobotTaskFacade

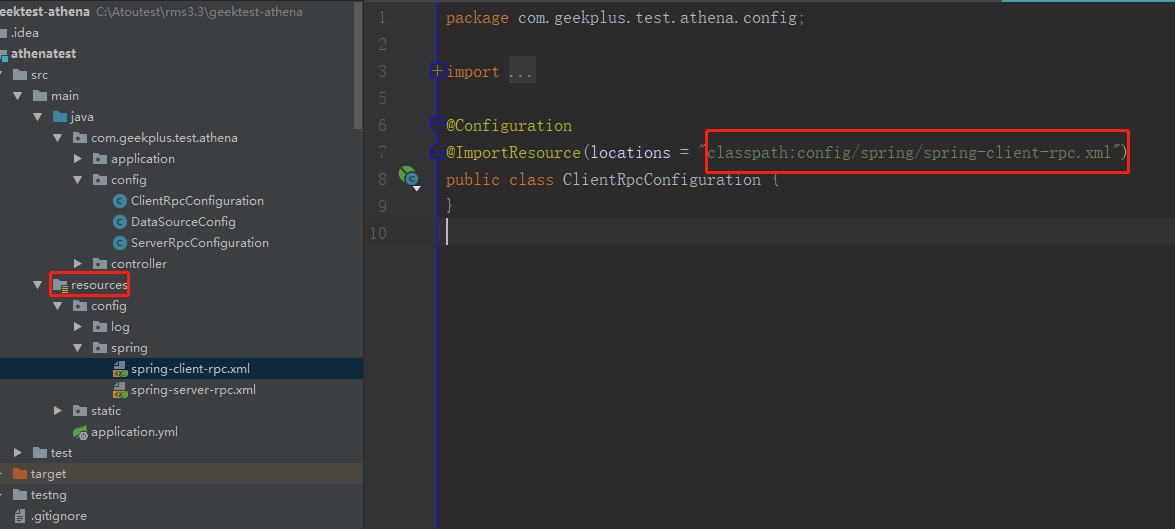
2，定义服务端（业务把自己当作服务，因为要接收rms调用callback接口，实现它）



1. beanid就是RobotTaskCallbackFacadeImpl实现类的名字
2. 具体实现是RobotTaskCallbackFacadeImpl实现了RobotTaskCallbackFacade
3. 自己RobotTaskCallbackFacadeImpl接口的端口号是10001

总结：

1. 无论是客户端，还是服务端，都必定义相同的定义公共接口例：RobotTaskCallbackFacade
2. 被调用方必须实现公共定义的接口例：RobotTaskCallbackFacadeImpl
3. 其实都可以放在一个xml定义，
4. 把定义的接口，或实现的接口作为bean加入到容器的方法，定义配置类



\*Configuration：表明配置类，加入bean的作用

\*@ImportResource(locations = "classpath:config/spring/spring-client-rpc.xml")把定义好的xml中bean加载进来，路径为类路径classpath，从resources开始写

5．使用bean；（加载到上下文，使用实例化，两种用法）

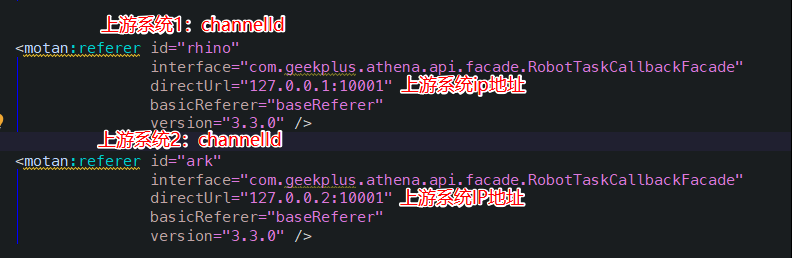
5.1可以使用@autowire自动加载到上下文中

5.2可以自定义一个类实现ApplicationContextAware()接口（也就是在实现ApplicationContextAward得类实例化后会调用setApplicationContext（）,即下文可以实现实例化后给对象属性**Map<String, RobotTaskCallbackFacade>赋值**），

重写setApplicationContext(ApplicationContext var1)方法，可以获得任意已经装载进容器的bean,通过class<T>即stundent.class 如下

**@Component("CallbackRPC")  
public class RPCCallback implements Callback, ApplicationContextAware {  
 */\*\*  
 \* 用来接收bean集合map类型，属性名为beanid,属性值为实际的bean类型  
 \*/* private Map<String, RobotTaskCallbackFacade> robotTaskCallbackMap;  
 */\*\*  
 \* 新机器人任务回调接口  
 \*/* private Map<String, WarehouseCallbackFacade> warehouseCallbackFacadeMap;**

**//根据bean类型获取bean集合（可能同一个类或接口被定义为多个bean,例如多rpc时，）**



**@Override  
public void setApplicationContext(ApplicationContext applicationContext) throws BeansException {  
 robotTaskCallbackMap=applicationContext.getBeansOfType(RobotTaskCallbackFacade.class);  
 warehouseCallbackFacadeMap=applicationContext.getBeansOfType(WarehouseCallbackFacade.class);  
}**

**遍历使用bean**

**for (RobotTaskCallbackFacade facade : robotTaskCallbackMap.values()) {**

**map.valuse()---注：获取集合的所有values** **Collection<V>类型**

**map. keySet ()---注：获取集合的所有keys** **Set<K> keySet()类型**

**, }**