Rúben Santos - 47766

Alexandre Santos - 47926

Ana Pereira - 47930

November 7, 2020

1ªAula Prática

Java RMI

# Exercício 1

No primeiro exercício é pedido que se desenvolva uma aplicação Java RMI cliente que permita ao utilizador jogar um jogo de pesca presente numa máquina virtual de um servidor da Google Cloud Platform.

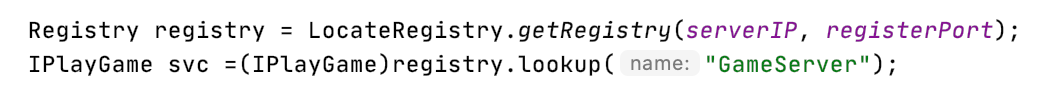
A comunicação entre cliente servidor é possível através do protocolo Java RMI Registry em conjunto com a integração de uma biblioteca contracto nos dois pontos.

A biblioteca que servirá de contracto é um ficheiro ”.jar” e é através dele que será possível ao cliente instanciar classes e executar métodos implementados no lado do servidor.

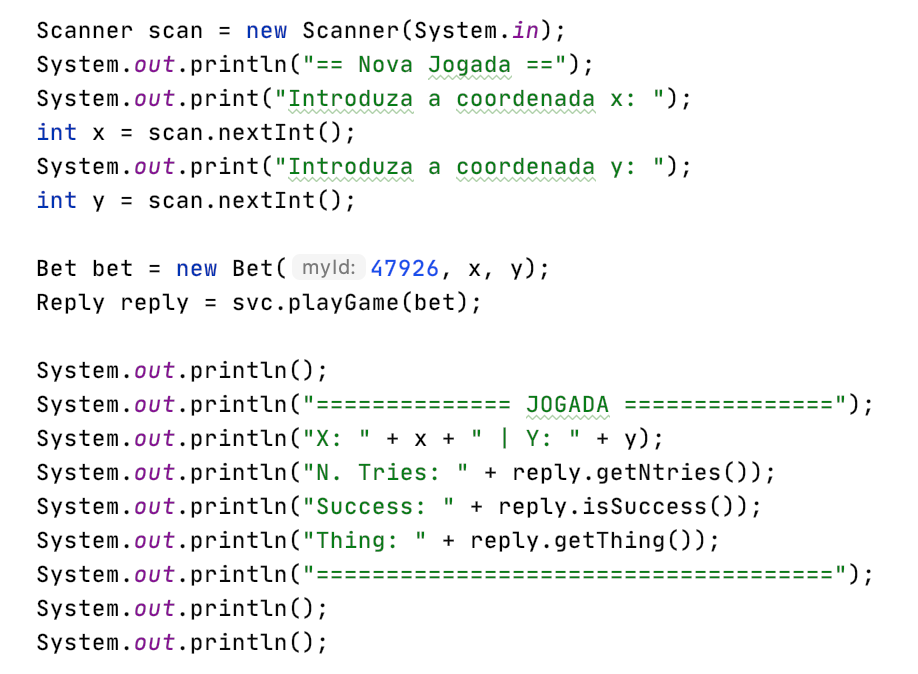
No lado do cliente, após a integração do contracto no projeto é necessário instanciar um objeto Registry que será apontado ao servidor usando como recurso o endereço de ip e porto disponibilizados no enunciado do exercício.

Através desse objeto criado, e com uma comunicação disponível estre os dois pontos, vai ser possível instanciar um objeto “IPlaygame” que representa a interface pela qual serão feitas as jogadas.

Ao fazer esta instanciação é necessário definir o nome do objeto registado no rmi registry do lado do servidor, neste caso “GameServer”, também ele disponibilizado no enunciado.



A partir deste ponto, a comunicação está estabelecida e é possível receber e enviar informação entre os dois pontos através das classes e respetivos métodos presentes no contracto.



Neste caso em concreto, é pedido ao utilizador que introduza as coordenadas X e Y. Estas coordenadas são usadas na instanciação de um objeto “Bet” que será usado como parâmetro no método “playGame”. Este, por sua vez, retorna um objeto “Reply” que contém a informação resultante da jogada proveniente do servidor.

A Finalidade do jogo é encontrar 3 perolas escondidas num rio fazendo apostas nas suas coordenadas.

As 3 perolas foram encontradas nas seguintes coordenadas:

* X: 2 – Y: 1
* X: 4 – Y: 2
* X: 6 – Y: 0

# Exercício 2

Primeiramente foi criado um projeto auxiliar no qual definimos o objeto e as duas interfaces do contrato. *ILeiloes e INotification.* Deste projeto foi executado um artefacto *jar*, que será usado posteriormente em ambas aplicações *cliente* e *servidora.*

A aplicação cliente deverá implementar a interface INotification e deverá disponibilizar o objeto que implementa essa interface, através do mecanismo Java RMI, a implementação do INotification será posteriormente usada pela aplicação servidora como um *callback*.

A aplicação servidora, implementa a interface ILeiloes. A implementação desta interface passa na nossa ótica pela criação de um sistema de registo de licitações, usando mecanismos Java RMI para disponibilizar a implementação da interface ILeiloes para a ou as aplicações cliente que quiserem usufruir do serviço do ILeiloes. Foi criado um projeto que simula três clientes, cada um deles gerando um objeto RMI em portas distintas (*INotification)*.

O serviço do ILeiloes, inicia o registo de um objeto (*SomeObject)* quando recebe uma execução do ***initLeilao*** de uma aplicação cliente, a aplicação servidora guarda numa mapa ( representação de uma base de dados) onde a *key* é o id do objeto que é uma propriedade do objeto *SomeObject* e como *value* guarda um objeto interno que guarda um valor da licitação ( que começará a zero) e o objeto *INotification* que será usado posteriormente para notificar o cliente. Um cliente pode solicitar a devolução de todos os objetos guardados no mapa através do método ***getAllLeiloes.*** Na implementação do ***licitar*,** o serviço executa a atualização do valor da licitação associado ao objeto pedido e executa através do *callback* recebido a notificação da licitação realizada; posteriormente, o serviço verifica todos as licitações e executa os *callbacks* guardados em initLeilao*,*de forma a notificar os outros clientes da atualização da licitação para o objeto que acabou de sofrer alteração. Em baixo, uma imagem que mostra os *logs* da execução do cliente vs servidor na mesma máquina física.

Text

Description automatically generated