Estácio

Por que não paralelizar

Nome: Thomaz Alves Da Costa **Matrícula:** 202303196751

Campus: Campus Virtual (SIA)
Por que não paralelizar – 2023.1 – 3º Semestre

Objetivo da Prática:

- Criar servidores Java com base em Sockets.
- Criar clientes síncronos para servidores com base em Sockets.
- Criar clientes assíncronos para servidores com base em Sockets.
- Utilizar Threads para implementação de processos paralelos.
- No final do exercício, o aluno terá criado um servidor Java baseado em Socket, com acesso ao banco de dados via JPA, além de utilizar os recursos nativos do Java para implementação de clientes síncronos e assíncronos. As Threads serão usadas tanto no servidor, para viabilizar múltiplos clientes paralelos, quanto no cliente, para implementar a resposta assíncrona.

1º Procedimento | **Criando o Servidor e Cliente de Teste**

Resultados do procedimento 1:

a) Como funcionam as classes Socket e ServerSocket?

Socket**:** uma conexão de comunicação bidirecional entre dois programas.

ServerSocket: usado pelo servidor para aguardar e aceitar conexões de entrada de clientes.

b) Qual a importância das portas para a conexão com servidores?

As portas são fundamentais para que os dados sejam encaminhados entre o cliente e o servidor.

c) Para que servem as classes de entrada e saída ObjectInputStream e ObjectOutputStream, e por que os objetos transmitidos devem ser serializáveis?

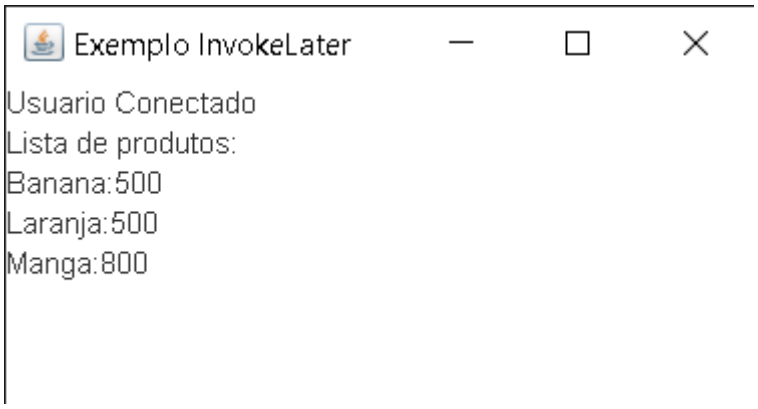
São usados para serializar (ObjectOutputStream) ou desserializar (ObjectInputStream) uma sequência de bytes.

d) Por que, mesmo utilizando as classes de entidades JPA no cliente, foi possível garantir o isolamento do acesso ao banco de dados?

Utilizar classes de entidades JPA (Java Persistence API) no cliente não garante por si só o isolamento do acesso ao banco de dados. A JPA é uma especificação Java para mapeamento **objeto-relacional,** que permite aos desenvolvedores mapear objetos Java para tabelas de banco de dados e vice-versa de forma transparente.

2º Procedimento | **Servidor Completo e Cliente Assíncrono**

Resultado do procedimento 2:



a) Como as Threads podem ser utilizadas para o tratamento assíncrono das respostas enviadas pelo servidor?

Podem ser utilizadas para o processamento e recebimento de mensagens assíncronas.

b) Para que serve o método invokeLater, da classe SwingUtilities?

Serve para executar determinada tarefa de forma assíncrona.

c) Como os objetos são enviados e recebidos pelo Socket Java?

São enviados através de sockets serializados ou desserializados.

d) Compare a utilização de comportamento assíncrono ou síncrono nos clientes com Socket Java, ressaltando as características relacionadas ao bloqueio do processamento.

Síncrono: o Componente síncrono bloqueia o processamento, ou seja o cliente fica bloqueado até que a resposta seja recebida.

Assíncrono: Já o Componente Assíncrono não há o bloqueio de processamento, permitindo que o cliente continue a sua execução até que receba a resposta.

Conclusão:

Criar um sistema paralelo em java é bem legal, pois podemos criar a parte do server e a parte do cliente, no qual o cliente envia um comando para o servidor e o servidor devolve uma resposta, assim podemos manipular a resposta do servidor no cliente, porém existe algumas dificuldades em questão de parar uma thread, visto que mesmo que ela saia do loop, o sistema continua a roda, assim então tendo que fazer algo para interromper de forma controlada.