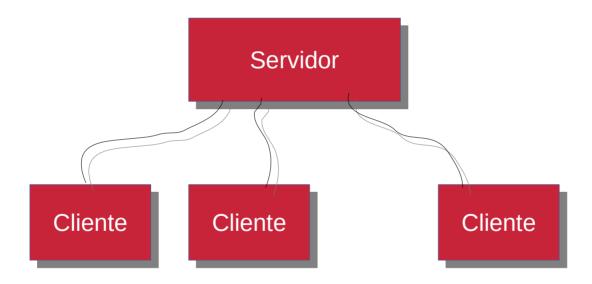
Sockets e Multithread

Estudo dirigido

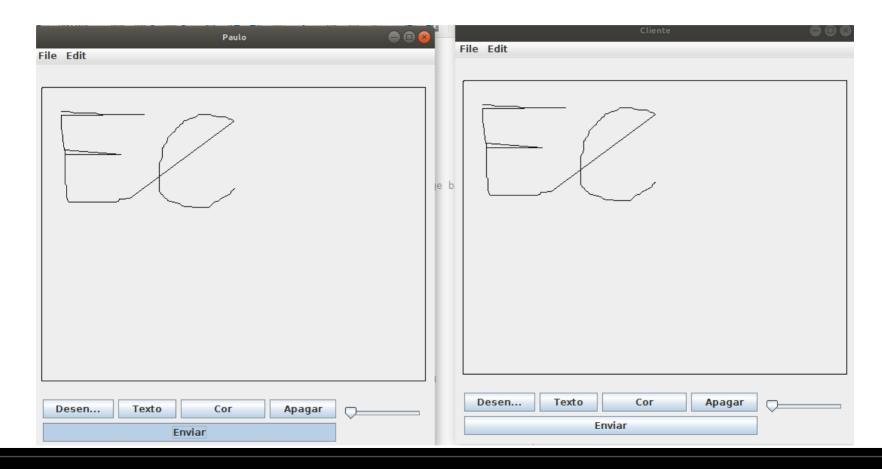
Objetivos

- Implementar um sistema multithread que explore os recursos de interface gráfica e sockets.
- Apresentar sistemas computacionais mais complexos preparando os discentes para a realidade das empresas.
- Proporcionar experiências com Threads.

Projeto



Projeto



Classes

- DeitelMessagerMagicTable
- ▼ 🚭 Source Packages
 - ▼

 com.deitel.messenger
 - CanvasConstants.java

 - CommandListener.java
 - MessageListener.java
 - MessageManager.java
 - MyCanvas.java
 - SwingControlDemo.java
 - 🔻 🖺 com.deitel.messenger.images
 - Connect.gif
 - Disconnect.gif
 - Send.gif
 - com.deitel.messenger.sockets
 - SocketMessengerConstants.java
 - E com.deitel.messenger.sockets.client
 - DeitelMessenger.java
 - MessageSender.java
 - PacketReceiver.java
 - SocketMessageManager.java
 - E com.deitel.messenger.sockets.server
 - DeitelMessengerServer.java
 - DeitelMessengerServerTest.java
 - MessageReceiver.java
 - MulticastSender.java

CommandListener → Mensagens Canvas e GUI.

MessageListener → Tratar os dados recebidos.

MessageManager → Enviar e receber dados.

MyCanvas → Painel para desenho.

SwingControlDemo → GUI do Cliente.

Pacote client

Message Sender → Formata a mensagem e escreve para o servidor.

PacketReceiver → Recebe a mensagem do servidor e utiliza a implementação da interface MessageListener para tratar os Dados.

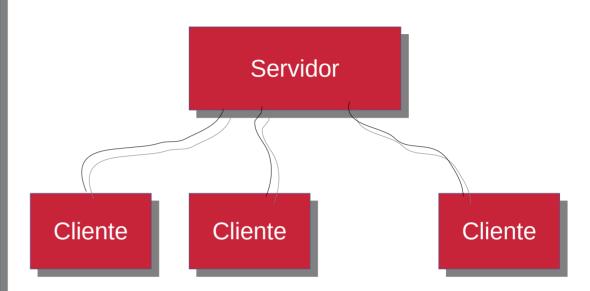
SocketMessageManager → responsável pela comunicação cliente → servidor (socket).

Pacote Server

MessageReceiver → Recebe as mensagens enviadas pelos clientes. Utiliza a a implementação de MessageListener para tratar os dados recebidos. Encaminha os Dados para a classe MulticastSender.

MulticastSender → Recebe a mensagem tratada e envia em multicast para todos os Clientes conectados.

- 1. Servidor executa e escuta por Conexão.
- 2. Cliente conecta no servidor e Aguarda receber um pacote.
- 3. Servidor bloqueia aguardando Mensagem.
- 4. Cliente envia a mensagem.
- 5. Servidor recebe a mensagem.
- 6. Servidor processa a mensagem.
- 7. Servidor envia a mensagem a Todos clientes conectados.
- 8. Cliente recebe a mensagem
- 9. Cliente processa a mensagem.
- 10. Cliente bloqueia aguardando um Novo pacote.
- 11. Tudo isso em threads separadas De tal forma que o usuário pode Interagir com GUI.



Projeto - Especificações

- Ajustar o protocolo para receber cores, tamanhos de linhas, textos e apagar.
- Melhorar as Threads para evitar travamentos da interface gráfica.
- Remodelar a classe GUI (cliente). Definir uma nova identidade visual com menus, ícones.
- Documentar no código as alterações feitas.
- Máximo de 2 alunos por projeto.