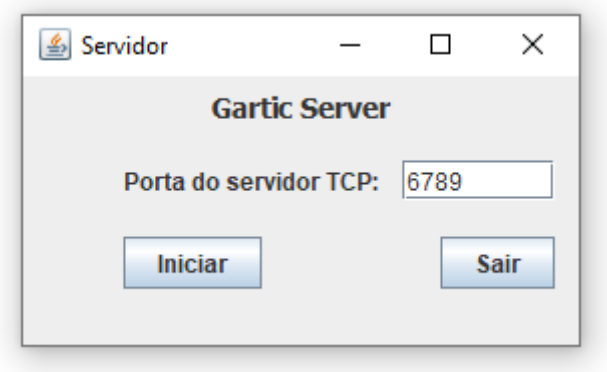
Nomes: Gabriel Montrazi Manduchi RA: 201601143
Guilherme Maioli RA: 201601121

Objetivo
O objetivo deste trabalho é realizar um jogo de Gartic. O jogo tem o propósito de um *player* (cliente) desenhar e o outro *player* (cliente) visualizar o que está sendo desenhado e então, tentar adivinhar o que está sendo desenhado em sua tela. A interação ocorre via servidor, o cliente que está desenhando envia o seu desenho e em quais coordenadas, enquanto o servidor manda as coordenadas para os clientes e pinta elas. Se o cliente que palpita acerta o desenho, ele se torna o 'desenhista', e o antigo 'desenhista' se torna o 'palpitador'.

Projeto

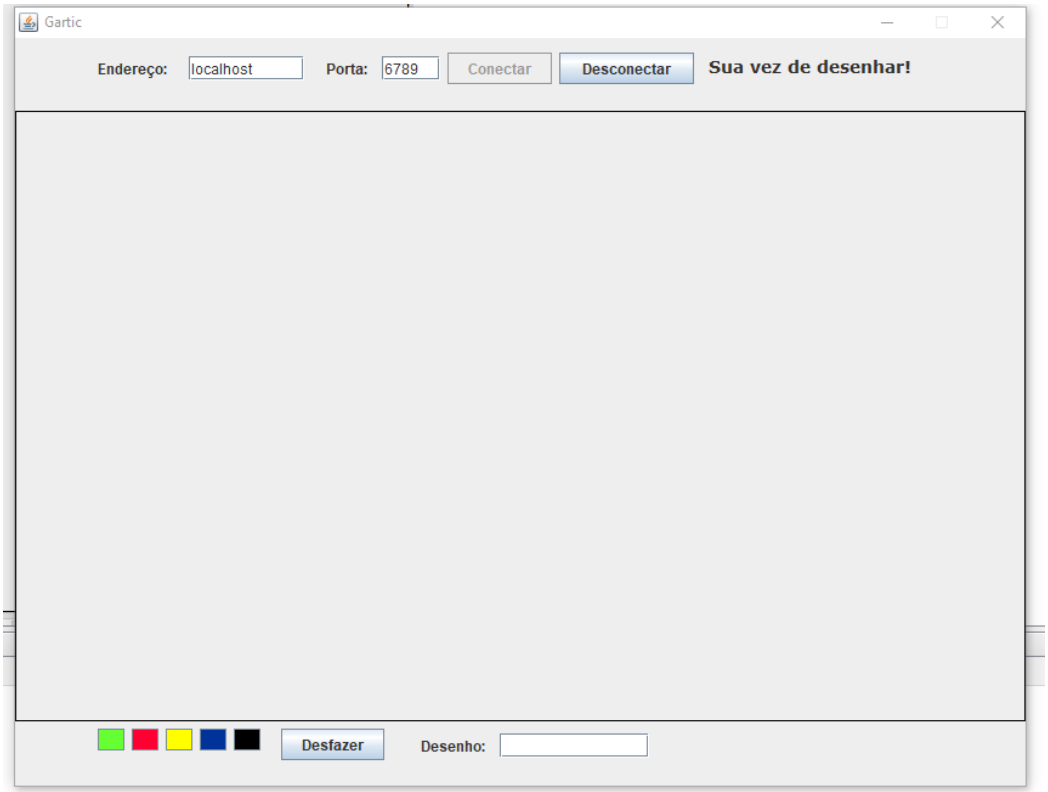
Interface
A interface visual do lado do servidor, é uma interface simples onde o usuário apenas informa a porta que ele utilizará para realizar a comunicação do jogo e inicia o serviço do servidor.



Já a interface visual do cliente, contém várias opções para o usuário e consequentemente cada opção troca uma mensagem diferente entre clientes e servidor.

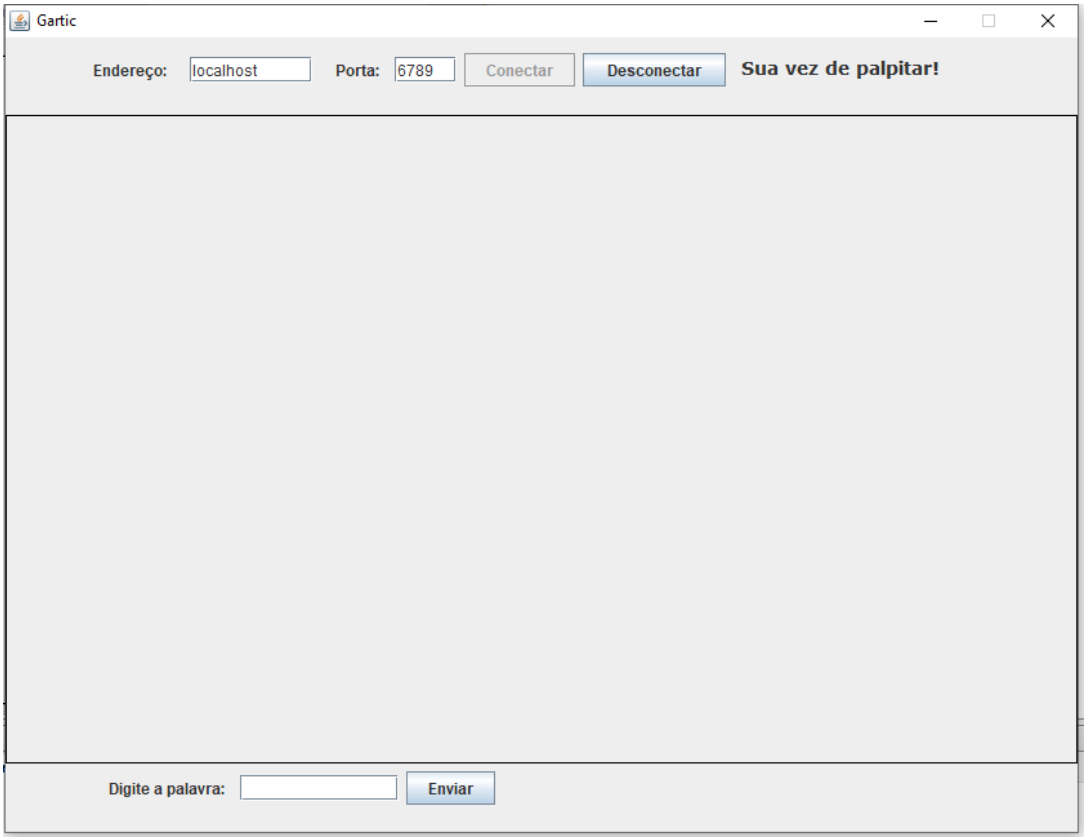
Existem duas telas diferentes para os clientes. A primeira é para o cliente desenhista, isto é, o cliente que irá realizar o desenho para seu adversário tentar acertar, a interface desenhista pode ser vista abaixo.

Cliente 'desenhista'



A segunda interface, ilustrada abaixo, é para o cliente palpitador. Ele só terá a opção de inserir no campo de texto o que acha que é o desenho.

Cliente 'palpitador'



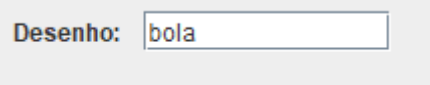
Troca de mensagens e funcionamento da aplicação:
O protocolo de comunicação usado na troca de mensagens de toda a aplicação foi o protocolo TCP.

Inicialmente quando um novo cliente se conecta, é criado um objeto nomeado no projeto como *ServerConnection* que representa o cliente em nossa aplicação. Na primeira criação, uma variável do cliente definida como *isFirst* é atribuída como *true*, isso ocorre pois o primeiro cliente a se conectar é o primeiro a iniciar desenhando.

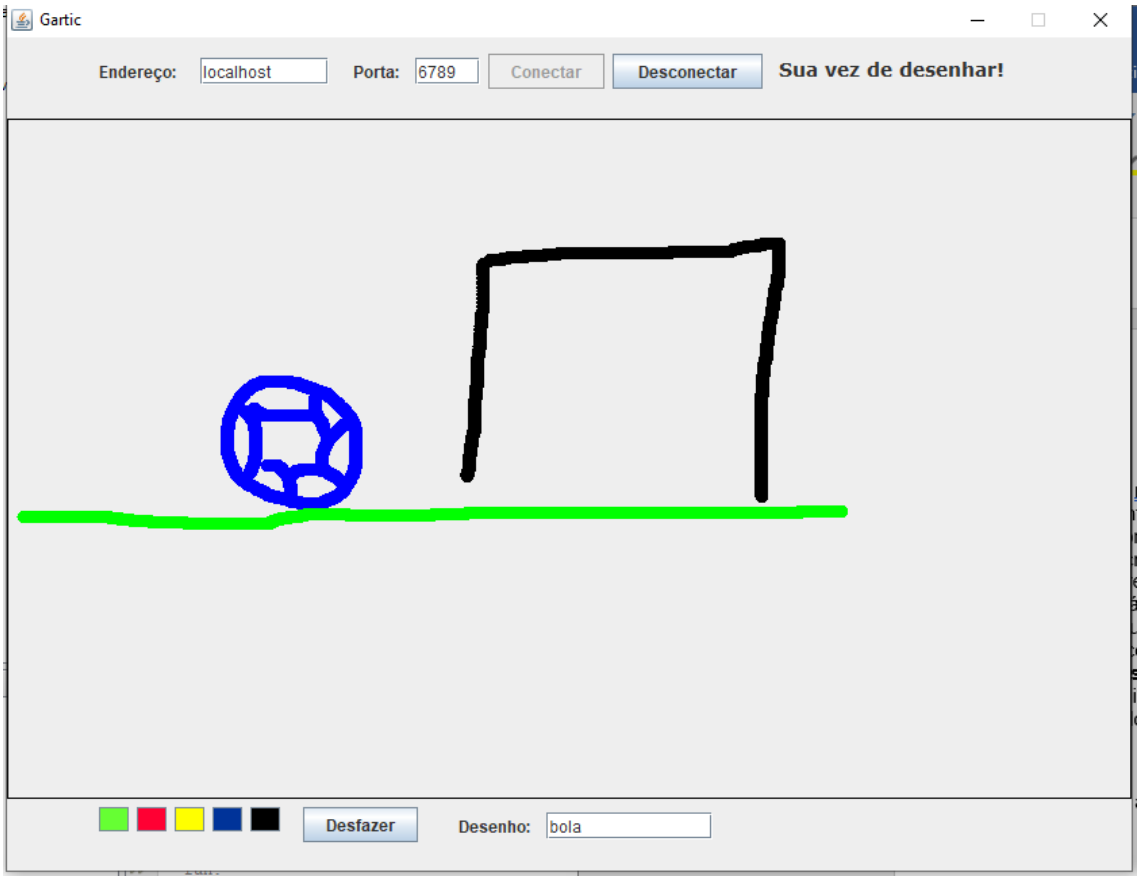
Essa variável também realiza o controle de quem está desenhando, o desenhista sempre terá esta variável como *true*, já o player que está palpitando, terá esta variável como *false*.

E ao desconectar 'matamos' esse objeto criado para aquele cliente.

-Desenhista:
Para o desenhista iniciar o desenho ele terá que informar primeiramente o que será desenhado no campo:



Com isso a interface libera o painel de desenho para iniciar o desenho, a partir disso, ele começa a desenhar.



O desenho ocorre toda vez que clicamos na tela e arrastamos ou apenas clicamos.
Ao realizar um clique, o cliente manda a seguinte mensagem para o servidor:

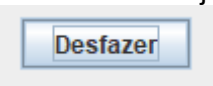
“1 | posição x | posição y | tamanho do pincel | cor | desenho”

-1: padrão atribuído para sempre que a troca de mensagem for para um desenho.
-Posição X: posição da coordenada X que você está clicando naquele momento na tela.
-Posição Y: posição da coordenada Y que você está clicando naquele momento na tela.
-Tamanho do pincel: não colocamos opção de escolha, mantemos um valor padrão de 10.
-Cor: cor que o desenho está sendo realizado naquele momento.
-Desenho: o desenho que está sendo feito.

A mensagem ao chegar no servidor, é guardada em três listas, uma que contém todas posições X já pintadas, outra as Y e outra as cores pintadas. ListaX[i], ListaY[i] e ListaCor[i] pertence a um pixel pintado na interface.

Também guardamos no servidor o desenho que está sendo realizado naquele momento. Exclusivamente neste caso, a mesma mensagem com o mesmo formato é enviada a todos os clientes (palpitadores e desenhistas). E na classe *ClientHandller* é realizado a atribuição das variáveis e chamadas de métodos para os desenhos serem realizados em todos os clientes.

O desenhista tem a opção de apagar um desenho já realizado, com o botão:



Ao clicar neste botão o cliente envia para o servidor a seguinte mensagem:

“apaga”

O servidor ao identificar a chegada desta mensagem, consulta os últimos elementos presente na lista de posição e monta uma mensagem:

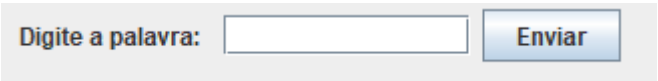
“apaga | posição X | posição Y | cor”

-Apaga: mensagem indicativa que algo será apagado.
-Posição X: posição da coordenada X que será apagada.
-Posição Y: posição da coordenada Y que será apagada.
-Cor: cor que será apagada.

Antes do servidor enviar a mensagem, ele apaga o último elemento de ambas listas e envia a mensagem para todos os clientes.

Ao chegar no *ClientHandler,* ele atribui a variáveis da interface gráfica e chama método que realiza a exclusão do último pixel desenhado.

Palpitador:
O cliente classificado como palpitador tem o objetivo de acertar o que está sendo desenhado em sua tela. Ele realiza esse palpite pelo campo:



Ao clicar em enviar o cliente manda uma mensagem para o servidor com o seguinte formato:

“palpite | o que ele acha que é”

-Palpite: para indicar que a mensagem é um palpite do cliente.
-O que ele acha que é: é o que ele acha que está sendo desenhado na tela.

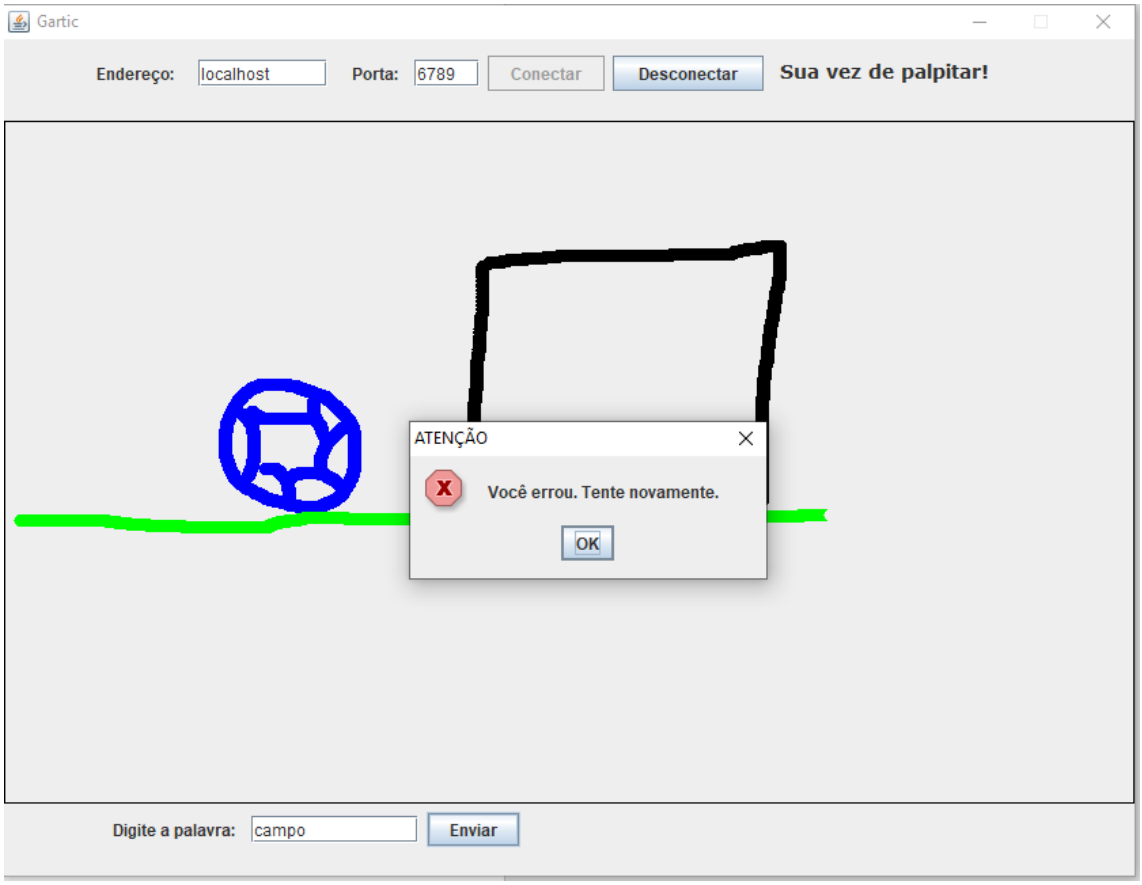
A mensagem ao chegar no servidor é realizada uma verificação, o servidor sabe o que está sendo desenhado então ele verifica se o palpitador está certo ou não.

Caso o servidor identifique que o palpitador errou, ele envia só para o palpitador uma mensagem:

“3”

-3: forma proposta para indicar que o palpite está ERRADO.

Com a chegada da mensagem “3” no *ClientHandller*, ele chama um método que indica que o cliente errou, e esse método mostra uma mensagem na tela apenas para aquele cliente.

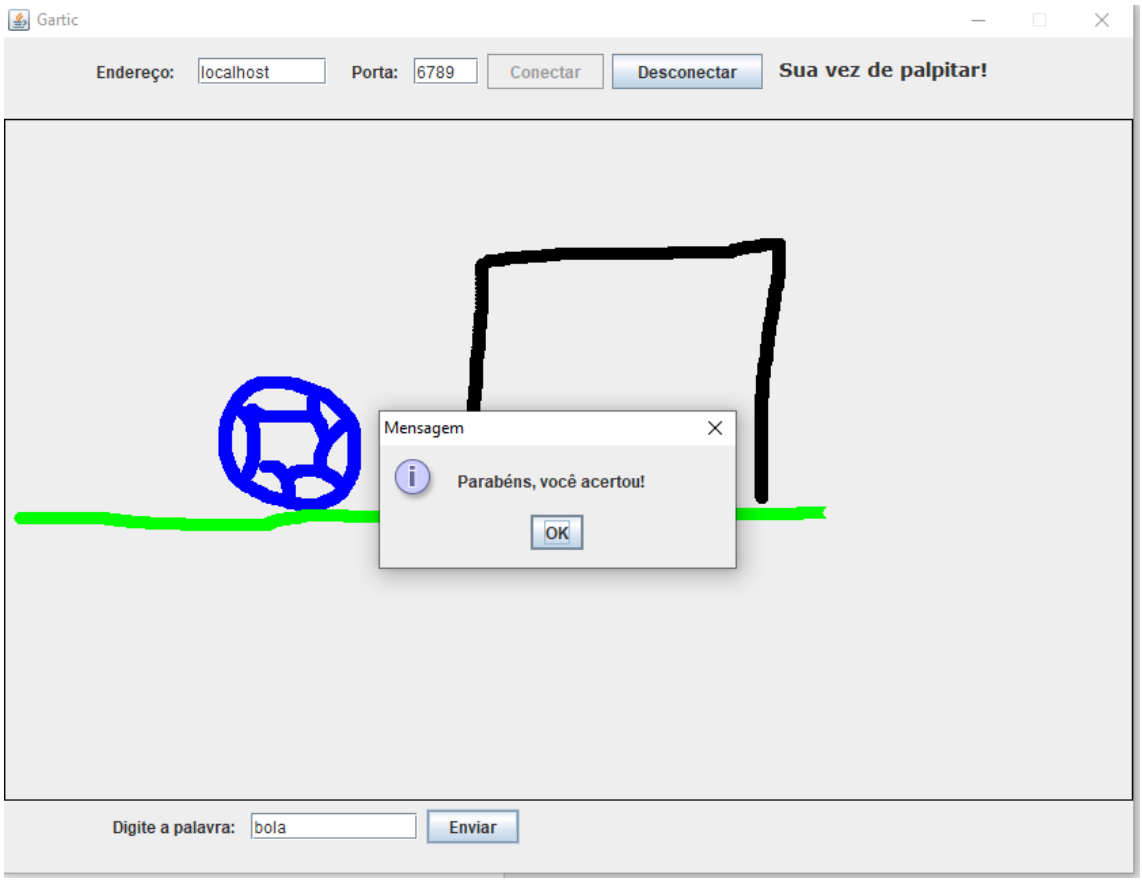


Caso o servidor identifique que o palpitador acertou, ele envia uma mensagem:

“2”

-2: forma proposta para indica que o palpite está CERTO.

Com a chegada da mensagem “2” no *ClientHandller*, ele chama um método que indica que o cliente acertou, e esse método mostra uma mensagem na tela apenas para aquele cliente.



Ao clicar no “Ok” desta mensagem de acerto, o cliente reenvia uma mensagem ao servidor:

“acertou”

-Acertou: forma escolhida pelo autor de avisar ao servidor para ele realizar a troca de cliente, isto é, quem era o desenhista se torna palpitador e quem era palpitador se tornará desenhista.

A mensagem ao chegar no servidor, primeiramente troca as variáveis do *isFirst* dos clientes, que indica quem desenha, após isso monta uma mensagem da seguinte forma:

“ganhou | listaX[1] | listaY[1] | listaX[2] | listaY[2] | ... | listaX[n] | listaY[n]”

-Ganhou: variável que indica que o jogo terminou.
-ListaX[n]: todas as posições de X desenhada.
-ListaY[n]: todas as posições de Y desenhada.

Antes de enviar a mensagem a todos os clientes para limpar suas telas, é realizado a limpeza das três listas e também a limpeza da variável desenho.

Ao chegar essa mensagem no *ClientHandler*, os clientes leem a mensagem inteira apagando as posições desenhadas na tela. E após isso é trocado a variável da tela que indica se o cliente é um desenhista ou palpitador. Com isso é chamado o método para liberar opções referente ao o que o cliente é desenhista ou ao palpitador.

Com isso voltamos a troca de mensagem inicial já descrita, que é quando o cliente realiza um desenho.

Instalação e teste:
Basta executar inicialmente o Server.java e escolher uma porta.

Após isso executar o Client.java com o mesmo IP do server e mesma porta.

- O primeiro Client executado começara desenhando
- O segundo Client começara palpitando

Ao realizar o acerto do desenho a aplicação automaticamente reiniciará, porém o cliente que estava desenhando se tornará o cliente que irá palpitar e o palpitador tornará o desenhista.

Observação: A aplicação foi desenvolvida apenas para dois clientes simultâneos.