UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Curso de Ciência da Computação

Autoras: Anna Gabriele Marques de Oliveira (agmoliveira@inf.ufpel.edu.br)
Naomi Celestino Ribes (ncribes@inf.ufpel.edu.br)

Objetivo:

O objetivo principal da aplicação desenvolvida é proporcionar uma plataforma digital para jogar o jogo de cartas Uno com até 4 jogadores, utilizando o protocolo TCP (Transmission Control Protocol) para estabelecer e gerenciar a comunicação entre os participantes. O protocolo foi projetado para garantir uma experiência de jogo fluida, confiável e sincronizada, mantendo a fidelidade às regras do jogo e permitindo que os jogadores desfrutem do Uno através de conexões de rede.

Características do protocolo:

Optou-se por um modelo cliente-servidor, onde o servidor é responsável por coordenar o fluxo do jogo, manter a autoridade sobre as regras e garantir a sincronização do estado do jogo entre todos os clientes. O que proporcionou um controle centralizado e a capacidade de lidar com possíveis conflitos, como jogadas simultâneas ou discordâncias sobre o estado do jogo. O protocolo é sem estado, o que minimiza a complexidade e o consumo de recursos, permitindo uma implementação mais simples e escalável.

Máquina de estados cliente-servidor:

Máquina de estados servidor-cliente:

Mensagens cliente-servidor:

connect: Conecta um socket de cliente que usa o protocolo TCP a um socket de servidor que usa TCP.

send: É usado para enviar dados de um socket de cliente para um socket do lado do servidor e vice-versa, usa o protocolo TCP.

recv: Retorna os dados recebidos como objeto com bytes.

Como essas mensagens aceitam apenas strings, foi necessária a criação de diferenciações entre as strings. Dessa forma elas são do estilo: $TIPO/string$
Tal que, o tipo é subdividido em:

- YON: mensagens de sim ou não. Para os casos de se o usuário deseja iniciar a partida e deseja desafiar outro jogador.
- COL: mensagens para a questão de escolha de cor após uma carta especial.

- CAR: cuja string é do estilo $[hand|top|id]$ na qual hand é uma mensagem que descreve a mão que o jogador tem, a carta do topo atual, e o id do jogador atual. E faz a escolha da carta que será jogada.
- MSG: não possui efeito na jogabilidade, é para a impressão de mensagens.
- END: é para sair do fluxo das jogadas e limpeza da mão do jogador antes do encerramento do jogo.

Mensagens servidor-cliente:

send: É usado para enviar dados de um socket de cliente para um socket do lado do servidor e vice-versa, usa o protocolo TCP.

sendall: Envia dados para todos os clientes conectados, usa o protocolo UDP.

accept: Aceita uma solicitação de conexão recebida de um cliente TCP. Está presente tanto no lado do cliente quanto no servidor.

Formato das mensagens:

connect: Inicia uma apresentação de três vias (3-way handshake) com o socket do servidor e estabelece a conexão. Envia como parâmetro o endereço IP do cliente.

recv: Pode ser usado para receber dados de sockets TCP e UDP, tanto no lado do cliente como no do servidor.

send: Recebe uma string e envia os bytes do dado. Por questões de segurança e boa prática, recomenda-se que seja utilizado com o método encode() que converte a string de em Bytes antes do envio.

sendall: Diferencia-se de send() pela utilização do protocolo UDP e porque bloqueia a utilização de dispositivos de entrada e saída até que toda a informação seja enviada ou seja levantada uma exceção. Assim como no UDP não há como saber quantos dados, se houverem, foram enviados com sucesso.

accept: Quando é chamada no lado do cliente (recebendo o endereço IP e o número do socket do servidor) a solicitação de conexão é recebida com a chamada accept() no lado do servidor, que retorna um socket que está conectado ao cliente.

<u>Estes formatos serão particularmente utilizados pelo servidor para:</u>
- Aceitação de cada um dos jogadores.
- O envio da primeira carta do baralho.
- O envio da resolução das cartas especiais (skip, +2 e +4) com exceção do reverso.
- Envia a atualização da mão do jogador para ele, e a quantidade atualizada de cartas nessa mão para todos os jogadores.
- Envia a mensagem de finalização da partida após a vitória de um jogador.

<u>Estes formatos serão particularmente utilizados pelo cliente para:</u>
- Inicialização da conexão com o servidor.
- Enviar qual carta o jogador escolheu de sua mão.
- Avisa ao servidor que deseja começar a partida com os jogadores que estão na sala com uma resposta de sim/não.
- Avisa ao servidor que deseja desafiar o outro jogador a fim de descobrir se o mesmo

teria uma outra carta da cor solicitada com uma resposta de sim/não (desafio).

- Envia ao servidor para qual cor de carta deseja trocar quando foi jogada uma carta especial que permite mudança de cor.

Especificação:
Outros métodos também utilizados foram:

listen: O servidor fica pronto ouvindo para detectar alguma conexão TCP. Possui como entrada o número máximo de conexões cliente-servidor que podem ser estabelecidas para esse socket pelo sistema operacional.

bind: Atribui um endereço IP e um número de porta a uma instância de socket. É usado quando precisamos de um socket de servidor, uma vez que os clientes devem ter ciência de qual socket é o do servidor.