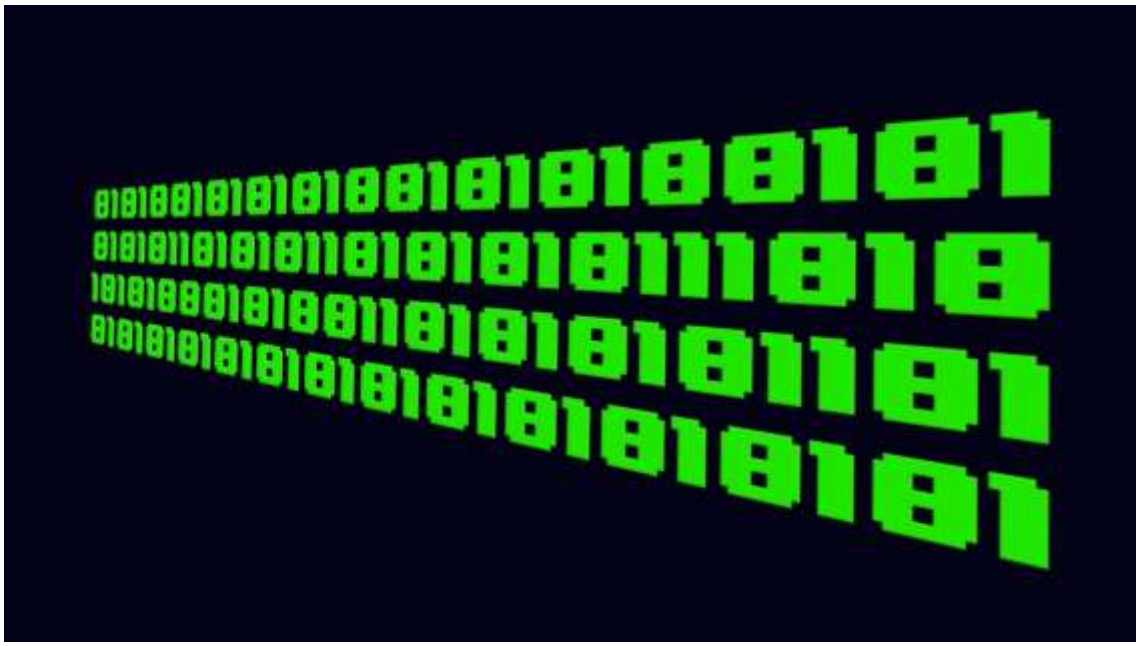
Redes de Computadores

Trabalho 2015/2016

Chat de Redes

UNIVERSIDADE DE ÉVORA



Realizado por:
João Calhau - 31621
José Pimenta - 31677

Indice

Introdução

Este trabalho enquadra-se na disciplina de Redes de Computadores , e sendo um trabalho final tem como finalidade a aplicação dos conhecimentos adquiridos ao longo desta disciplina.
Neste trabalho iremos então implementar um Chat de Redes, para isso iremos implementar o cliente e em principio um bot que joga com clientes.
Este relatório irá explicar os comandos que o cliente pode executar no chat, as informações enviadas do cliente para o servidor, do servidor para o cliente, assim como as decisões que foram tomadas ao longo da realização deste trabalho.
Iremos assim tentar dar o nosso melhor, e iremos tentar utilizar os métodos que achemos mais correctos ou propícios à boa evolução do trabalho e que no final se concretize o que nos é pedido.

Cliente Simples

O ficheiro cliente é o ficheiro principal do trabalho, é o ficheiro que o utilizador utiliza para contactar outros clientes e jogar jogos contra bots, caso estes existam.
O ficheiro cliente divide-se em 2 grandes partes, a parte em que o cliente envia mensagens ao servidor e a parte em que o cliente recebe mensagens do servidor.
Na 1ª parte, o utilizador escolhe 1 ação a ser realizada e conforme a sua escolha, assim é pedido ao utilizador que insira através de input informação necessária para a realização dessa dita ação.
Na 2ª parte, o utilizador recebe informação vinda do servidor, que conforme o que esta seja, assim serão executados os devidos prints.
Iremos entrar nas páginas seguintes em detalhe em ambas as partes e as funções que são utilizadas em cada uma das partes.

Enviar Informação

Situada dentro do main do ficheiro cliente.py , é onde se encontram os prints iniciais a informar o utilizador que este se conseguiu conectar ao servidor ou não. Após isso é executada a função startChat()

startChat()

Ao executar esta função, é pedido um input ao utilizador, mostrando uma mensagem a indicar que chegou ao nosso chat. Esse input é para o cliente fazer LOGIN no chat com uma conta que este tenha criado, ou caso não tenha já conta criada que crie uma conta com NOVO.
Nesta função são verificados todos os casos possíveis de erros e ainda os casos de os nicknames utilizarem apenas determinados simbolos e não exceder mais de 32 caracteres, entre outros.
Esta função é uma função que ocorre sempre que se abre o ficheiro cliente, isto para que o utilizador seja obrigado a ter um NICK para utilizar o chat.

Seguidamente são mostrados todos os comandos possiveis no chat, através de um print que identifica os comandos com números de 0 a 17.
O programa entra agora num ciclo infinito em que o utilizador tem de inserir um dos números referidos de modo a executar uma das ações que pretenda.
As ações têm como correspondência funções, que são as seguintes:

Menu de Opções - 0

Efectua apenas um print do menu referido anteriormente com os comandos possíveis, facilitando a aprendizagem dos comandos.

Alterar Nickname - 1

mudarNick()

Função que recebe como input um nickname que o cliente quer para substituir o atual. É verificado se o nickname cumpre as normas de caracteres válidos e se não contém mais do que 32 caractéres. É enviada mensagem ao servidor começando por “NICK“.

Registar Nickname - 2
registarNick()

Função que recebe como input o nickname que o cliente quer registar. O nickname tem de ser o que o utilizador está a usar no momento ou então acontece um erro estipulado no protocolo. O utilizador é questionado se pretende inserir um email ou não. Caso o utilizador o pretenda, é feito mais um input para que o utilizador insira. É depois pedido ao utilizado que insira a password que este pretenda. Conforme tenha sido inserido email ou não podem acontecer 1 das 2 possiveis mensagens para enviar ao servidor. É enviada mensagem ao servidor começando por “REGISTER “.

Recuperar Nickname - 3
recuperarNick()

Função que recebe como input um nickname da qual se pretende recuperar a password, para isso tem de ser inserido um nickname que esteja registado, ou então recebe-se mensagem a informar que o nickname não está registado. É enviada mensagem ao servidor começando por “RECOVER “.

Enviar Mensagem - 4
enviarMensagem()

Função que recebe como input o nickname do destinatário de uma mensagem ou o nome de uma sala. O utilizador é questionado se pretende enviar mensagem para um cliente ou para uma sala, de modo a saber diferenciar para quem enviar (isto porque o cliente antes inseriu um nome, e caso pretenda enviar mensagem para uma sala, a função trata de inserir o @ referente aos nomes das salas).
É enviada mensagem ao servidor começando por “MSG “.

Entrar numa Sala - 5
entrarSala()

Função que recebe como input o nome de uma sala na qual o utilizador pretende entrar. Caso a sala não exista, a sala é criada.
É enviada mensagem ao servidor começando por “ENTER “.

Sair de uma Sala - 6
sairSala()

Função que recebe como input o nome da sala da qual o utilizador pretende sair, isto porque o utilizador pode estar ao mesmo tempo em várias salas, necessitando de escolher qual pretende sair. O utilizador é questionado se pretende deixar uma mensagem de despedida, e conforme deixa mensagem de despedida ou não, assim é a mensagem enviada para o servidor.
É enviada mensagem ao servidor começando por “LEAVE “.

Alterar Tópico de uma Sala - 7
alterarTopico()

Função que recebe como input o nome da sala que pretende-se alterar o tópico. O utilizador é questionado se pretende escrever um novo tópico, caso pretenda, este tem de inserir o novo tópico no input, caso contrário o tópico é apagado, e a sala fica sem tópico.
É enviada mensagem ao servidor começando por “TOPIC “.

Lista Utilizadores de uma Sala - 8
listarUtilSala()

Função que recebe como input o nome da sala sobre a qual se pretende conhecer os utilizadores. É enviada mensagem ao servidor começando por “WHO “.

Listar Utilizadores de Todo o Chat - 9
listarUtilChat()

Função sobre a qual se pretende conhecer os utilizadores de todo o chat. É enviada mensagem ao servidor começando por “ULIST“.

Listar Todas as Salas e os seus Tópicos - 10
listarSalas()

Função sobre a qual se pretende conhecer as salas e os respectivos tópicos de todo o chat. É enviada mensagem ao servidor começando por “RLIST“.

Sair do Chat - 11
sair()

Função que recebe como input uma mensagem de despedida caso o utilizador assim o deseje. O cliente abandona assim o chat. É enviada mensagem ao servidor começando por “QUIT“.

Pedido para Enviar Ficheiro - 12
enviarFicheiro()

Função que recebe como input o utilizador destinatário, a quem se pretende enviar um ficheiro. Insere-se o nome do ficheiro caso este esteja na mesma pasta do ficheiro cliente.py ou o path do ficheiro. É verificado se é um ficheiro válido e o seu tamanho. É enviada mensagem ao servidor começando por “SENDFILE“.

Aceitar Recepção de Ficheiro - 13
aceitarFicheiro()

Função que recebe como input o identificador do ficheiro que se autoriza a receber. É enviada mensagem ao servidor começando por “ACCEPTFILE“.

Rejeitar Recepção de Ficheiro - 14
rejeitarFicheiro()

Função que recebe como input o identificador do ficheiro que se rejeita a receber. É enviada mensagem ao servidor começando por “REFUSEFILE“.

Enviar Parte de Ficheiro - 15
enviarParte()

Função que recebe como input o identificador do ficheiro que se pretende começar a enviar. É enviada mensagem ao servidor começando por “SENDPART “.

NOTA: esta função não foi devidamente implementada, pelo que o cliente não é um cliente completo.

Cacelar Transferência - 16 cancelarTransferencia()

Função que recebe como input o identificador do ficheiro que se pretende cancelar a transferência. É enviada mensagem ao servidor começando por “ABORTFILE“.

NOTA: esta função não foi devidamente implementada, pelo que o cliente não é um cliente completo.

Confirmar Envio Total de Ficheiro - 17 confirmarEnvio()

Função que recebe como input o identificador do ficheiro que foi totalmente recebebido. É enviada mensagem ao servidor começando por “ENDFILE“.

NOTA: esta função não foi devidamente implementada, pelo que o cliente não é um cliente completo.

Receber Informação

O servidor a qualquer momento pode enviar informações para o utilizador.
É na função *socket_read_thread(socket)* que se recebe data do servidor.
Caso esta seja a 1ª vez a receber informação, ou seja, informação recebida no inicio da ligação do servidor quando se pretende utilizar um NOVO nick, ou fazer LOGIN com um nick registado, é tratada de maneira diferente.
Nos restantes casos, na utilização normal do servidor, verifica-se se existe mensagem do servidor com terminal "\n" e se sim, é executada a função *mensagensServidor(alfa)*, em que alfa é uma mensagem do servidor pronta a executar. Conforme o que seja alfa, assim é executada a dita função.

mensagensServidor(temp)

Função que recebe temp como argumento. temp é uma mensagem do servidor em forma de lista, que tem separado por elementos, assim como a string original tinha espaços. Ou seja, fez um split da string pelos espaços e adicionou como elementos a uma lista, essa lista sendo temp.
Assim sendo, em todos os casos verifica-se qual o 1º elemento dessa lista.
Esse elemento pode ser um elemento que identifica a mensagem como informativa, de sucesso ou de erro.
Em todos os casos fazem-se prints adequados ao tipo de mensagem e utilizam-se outros argumentos da mensagem, se assim se verificar necessário.

Bot TicTacToe

Função "criarSala(socket)":

Simplesmente cria logga-se ao servidor com o nick temporário "botCenas" e cria uma sala chamada "TicTacCenas"

Função "checkLine(matrix)":

Verifica se algum jogador tem 3 jogadas na mesma linha (se um dos jogadores ganhou) e retorna True caso isso se verifique.

Função "checkColumn(matrix):

Faz o mesmo que a função "checkLine(matrix)" mas neste caso faz isso para as colunas.

Função "checkDiagonal(matrix)":

Faz o mesmo que a função "checkLine(matrix)" mas neste caso para as duas diagonais.

Função "checkDraw(matrix)":

Verifica se á um empate no jogo (se todas as posições da matriz estão preenchidas ou não), caso se verifique retorna True.

Função "botPlay(matrix)":

Função que faz a jogada do bot, neste caso um bot "burro" que joga em casas random.

Função "playerPlay(line, column, matrix)":

Função que recebe uma linha e uma coluna e faz a jogada do jogador na matriz consoante essa linha e coluna

Função "initialize()"

Inicializa a matriz de 3 x 3 e enumera as casa de 1 a 9.

Na Main:

Depois de tentar conectar-se ao server inicia-se um ciclo “While” que fica sempre a correr.

O primeiro if verifica se há alguma coisa para ler do array “command” que tem as mensagens do utilizadores a quererem jogar. Caso não haja nada continua o ciclo.

O primeiro elif verifica se algum utilizador entrou na sala. Visto que sempre que um utilizador entra na sala manda uma mensagem a todos os outros utilizadores a dizer que alguem entrou.

O segundo elif verifica se o comando lido é, em especifico, uma mensagem de um utilizador, verifica ainda se não é uma mensagem da sala.

Dentro do segundo elif é que começa o jogo propriamente dito, primeiro o bot verifica se o comando é um play, um move ou um show.

No caso do “play”, verifica se o jogador que mandou play já está num jogo ou não, acedendo ao dicionario de jogadores correntes, caso não esteja inicializa uma matriz nova a parir da função “initialize” e faz a primeira jogada (o bot joga sempre primeiro e é sempre o X), de seguida adiciona o novo jogador ao dicionario e envia um “print” do tabuleiro ao utilizador. Se o jogador já estiver a jogar manda uma mensagem a dizer o mesmo.

No caso do comando ser “move”, primeiro vai verificar se o utilizador já tem um jogo corrente, caso não tenha manda uma mensagem a referir o mesmo e continua o loop. Caso o jogador já exista no dicionario, o bot vai buscar a matrix corrente do jogador (guardada no dicionario como value) e verifica se o input do utilizador é um jogada válida, caso positivo o faz o jogador jogar na casa referida com a função “playerPlay”. O bot verifica antes de jogar se o jogo está empatado e procede. Depois do turno do Jogador, o bot verifica ainda se o jogo acabou, se o jogo tiver acabado a partir das funções “checkLine”, “checkColumn” e “checkDiagonal”, caso alguma delas se verifique quer dizer que o jogador ganhou, e neste caso o bot informa o jogar que ganhou através de uma mensagem e faz pop do jogador no dicionario para o mesmo poder jogar outra vez se desejar. Caso nenhuma delas se verificar o bot procede ao seu turno e joga. depois de jogar vai verificar outra vez com as 3 funções “check” se ele proprio ganhou, caso positivo informa o jogador através de uma mensagem e faz “pop” ao mesmo para poder jogar outra vez. Caso negativo atualiza o dicionario com a nova matriz.

Finalmente caso o comando seja “show”, primeiro verifica se o jogador tem um jogo corrente, caso positivo manda um print do tabuleiro, caso negativo informa o utilizador que não tem nenhum jogo começado.

Se o utilizador mandar algum outro comando diferente de “play”, “move” ou “show” o bot informa o utilizador que não sabe o que fazer com esse input.

Conclusão

Após a realização deste trabalho conseguimos aprender algumas coisas que não sabiamos ou não tinhamos tanta experiência. No geral foi um bom trabalho e divertido de fazer. Fizémos o cliente simples com alguns comandos de ficheiros feitos, mas sem puder enviar os ficheiros, do cliente completo, e fizémos um bot que joga o jogo do galo com vários clientes em simultâneo. Ficou assim um chat que pensamos que seja fácil de utilizar, e que poderia ser utilizado fora do contexto da disciplina como um cliente de chat funcional, simples e acessível para o utilizador.