|  |  |
| --- | --- |
| Brasão da UFC |  |

**Revisão Protocolos TCP e UDP**

**1 - Contextualização:**

A camada de transporte, tanto no Modelo OSI quanto no Modelo TCP/IP, é a camada responsável pela transferência de dados entre duas máquinas, independente da aplicação usada e do tipo, topologia ou configuração das redes físicas existentes entre elas. A camada de transporte reúne protocolos de transporte end-to-end entre máquinas, isto é, uma entidade (hardware/software) que utilize os protocolos desta camada só se comunica com a sua entidade destino, sem comunicação com máquinas intermediárias na rede, como pode ocorrer com as camadas inferiores. Dois dos principais protocolos desta camada são o UDP e o TCP.

A seguir são apresentados vídeos, leituras e fóruns de dúvidas que facilitarão seu entendimento sobre esse assunto. Você não precisa ver o conteúdo por completo, escolha os tópicos que sentir necessidade, de acordo com o tipo de mídia que você goste. Anote todas as possíveis dúvidas para levar à sala de aula no dia da revisão.

**2 - Vídeos**

|  |  |
| --- | --- |
| Tema | Link |
| [PT] Entenda os protocolos TCP e UDP (3:27) | <https://www.youtube.com/watch?v=cy-ITN-OODM> |
| [PT] Protocolos TCP e UDP (15:00) | <https://www.youtube.com/watch?v=uRvjPlbJ_98> |
| [PT] Modelo OSI e TCP/IP - Como funciona o processo de comunicação em redes (19:24) | <https://www.youtube.com/watch?v=oz8gvGIUKFw> |
| [EN] TCP and UDP: Comparing Transport Protocols (5:59) | <https://www.youtube.com/watch?v=MMDhvHYAF7E> |
| [EN] UDP and TCP: Comparison of Transport Protocols (11:34) | <https://www.youtube.com/watch?v=Vdc8TCESIg8> |

**3 - Leituras**

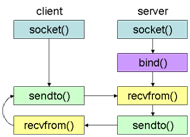
|  |  |
| --- | --- |
| Tema | Link |
| Internet: qual a diferença entre os protocolos UDP e TCP? | <https://www.tecmundo.com.br/internet/57947-internet-diferenca-entre-protocolos-udp-tcp.htm> |
| Entenda como Funciona o Protocolo TCP-IP | <https://www.citisystems.com.br/protocolo-tcp-ip/> |
| TCP - Slides (após slide 32 não é necessária a leitura) | <https://www.inf.ufes.br/~zegonc/material/Redes_de_Computadores/O%20Protocolo%20TCP.pdf> |
| Redes: Quais as diferenças entre o Protocolo TCP e UDP | <https://pplware.sapo.pt/tutoriais/redes-quais-diferencas-protocolo-tcp-udp/> |

**4 - Fóruns de Discussão**

|  |  |
| --- | --- |
| Tema | Link |
| O que é TCP e UDP? Qual a diferença entre os dois protocolos? | <https://pt.stackoverflow.com/questions/221860/o-que-%C3%A9-tcp-e-udp-qual-a-diferen%C3%A7a-entre-os-dois-protocolos> |
| Difference between TCP and UDP? | <https://stackoverflow.com/questions/5970383/difference-between-tcp-and-udp> |
| When would one use UDP over TCP? | <https://networkengineering.stackexchange.com/questions/32231/when-would-one-use-udp-over-tcp> |

**Introdução a Sockets TCP em Java**

Na atividade, usaremos a API de sockets para comunicar com um servidor de um jogo de corrida de cavalos. Durante a atividade, usaremos vários tipos de métodos de troca de dados entre os sockets. Para tal, você irá criar um cliente Java, no compilador que você preferir, para se comunicar com o servidor do jogo. Desta forma, antes de realizar a atividade é preciso revisar alguns conceitos básicos caso você não lembre.



A Figura acima ilustra o funcionamento de uma comunicação simples em TCP. Um servidor socket espera em uma determinada porta e quando um cliente realiza o bind um socket de comunicação é criado. O código a seguir ilustra a execução de um servidor na porta 9090. O servidor irá abrir a porta e ficará bloqueado na linha do “welcomeSocket.accept()” até que uma conexão seja estabelecida.

ServerSocket welcomeSocket = new ServerSocket(9090);

int i=0; //número de clientes

System.out.println ("Servidor no ar");

while(true) {

Socket connectionSocket = welcomeSocket.accept();

i++;

System.out.println ("Nova conexão");

Já para criar um cliente socket em Java é necessário instanciar um objeto da classe Socket

Socket clientSocket = new Socket("localhost", 9090);

O construtor do objeto socket permite que o primeiro parâmetro seja uma S*tring* que indica o endereço IP e o segundo indica a porta em que o socket vai se conectar no servidor.

Para a comunicação, nós podemos usar dois objetos, o de input e o de output. Eles representam formas mais simples de manipulação dos “canais stream” do sockets. O Java fornece uma ampla gama desses tipos de manipuladores de stream. Para começar, usaremos os mais comumente usados: o DataOuputStream e o DataInputStream.

DataOutputStream outToServer = new DataOutputStream(clientSocket.getOutputStream());

DataInputStream inFromServer = new DataInputStream(clientSocket.getInputStream());

O parâmetro do construtor de ambos *streams* são sockets clientes criados previamente.

Provavelmente, você já ouviu da forma mais básica de BufferedReader para ler Strings ao usar a sua função readLine(), contudo a comunicação do tipo Data permite que você envie qualquer tipo de informação e não limita ela a apenas uma linha de caracteres. Por exemplo, no código a seguir o cliente do socket está enviando a String digitada no teclado usando o método writeBytes e adicionado um “\n” a mensagem. Isso permite que do lado do servidor a mensagem possa ser lida usando o método readLine() do DataInputStream

clientSocket = new Socket("localhost", 9090);

DataOutputStream socketSaidaServer =

new DataOutputStream(clientSocket.getOutputStream());

BufferedReader socketEntrada =

new BufferedReader(new

InputStreamReader(clientSocket.getInputStream()));

System.out.println ("Digite a mensagem e tecle enter");

sentence = teclado.readLine();

socketSaidaServer.writeBytes(sentence + '\n');

socketSaidaServer.flush();

System.out.println ("Mensagem enviada");

modifiedSentence = socketEntrada.readLine();

System.out.println("Servidor: " + modifiedSentence);

clientSocket.close();

Outra forma possível é de enviar e receber mensagens de texto é usando os métodos writeUTF e readUTF como mostrados em sala de aula

Socket cliente = new Socket ("192.168.0.78",9090);

DataOutputStream streamSaida =

new DataOutputStream(cliente.getOutputStream());

streamSaida.writeUTF("Odeio código");

streamSaida.flush();

**Objetos de Stream em Java**

**1 Contexto**

A linguagem Java é capaz de processar arquivos que requisitam grandes quantidades de dados persistentes. Esses são os dados que ficam armazenados dentro dos arquivos, sendo que a sua duração vai além da finalização de um programa. Esse processamento de arquivos é uma capacidade que o Java oferece para **ler e gravar dados** na memória, arquivos e **conexões de rede**.

Quando existe um dispositivo de entrada de dados ativo, para obter dados deste dispositivo é usada uma ação conhecida como read(). E para enviar um dado para um dispositivo de saída é utilizado o método write().

Para trabalhar com entrada e saída de dados são utilizada as classes que estão dentro do java.io. Essas classes oferecem algumas funcionalidades como:

* Manipulação de entrada e saída de bytes – transferência de dados binários;
* Manipulação de entrada e saída de caracteres – transferência de textos;
* Buffers - melhoram a eficiência da velocidade de leitura e escrita;
* Conversão de formatos – texto e formato interno de dados binários.

As entradas e saídas de dados são manipuladas como sequência de bytes, através das classes **InputStream e OutputStream** e as classes dependentes.

A seguir são apresentados vídeos, leituras e fóruns de dúvidas que facilitarão seu entendimento sobre esse assunto. Você não precisa ver o conteúdo por completo, escolha os tópicos que sentir necessidade, de acordo com o tipo de mídia que você goste. Anote todas as possíveis dúvidas para levar à sala de aula no dia da revisão.

retirado de <https://www.devmedia.com.br/classes-de-entrada-e-saida-de-dados-em-java/26029>

**2 - Vídeos**

|  |  |
| --- | --- |
| Tema | Link |
| [PT] InputStream e OutputStream (18:00) | <https://www.youtube.com/watch?v=zvi6U0VxbjE> |
| [EN] Files and Streams (12:17) | <https://www.youtube.com/watch?v=3YRahx2ltSg> |
| [EN] BufferedReader, InputStream, BufferedInputStream: Overview (11:44) | <https://www.youtube.com/watch?v=onCa-1oPk2w> |

**3 - Leituras**

|  |  |
| --- | --- |
| Tema | Link |
| Classes de entrada e saída de dados em JAVA | <https://www.devmedia.com.br/classes-de-entrada-e-saida-de-dados-em-java/26029> |
| Classes InputStream e outputstream | <https://www.devmedia.com.br/classe-inputstream-e-outputstream-em-java/32007> |
| Java - Files and I/O | <https://www.tutorialspoint.com/java/java_files_io.htm> |
| IO Streams in Java – InputStream and OutputStream | <https://www.javafolder.com/io-streams-in-java/> |

**4 - Fóruns**

|  |  |
| --- | --- |
| Tema | Link |
| Qual a diferença entre usar FileInputStream e FileOutputStream ou Scanner e PrintStream? | <https://pt.stackoverflow.com/questions/83634/qual-a-diferen%C3%A7a-entre-usar-fileinputstream-e-fileoutputstream-ou-scanner-e-prin/83662#83662> |
| JAVA: qual a diferença entre OutputStream e getOutputStream()? | <https://pt.stackoverflow.com/questions/245114/java-qual-a-diferen%C3%A7a-entre-outputstream-e-getoutputstream?rq=1> |
| Easy way to write contents of a Java InputStream to an OutputStream | <https://stackoverflow.com/questions/43157/easy-way-to-write-contents-of-a-java-inputstream-to-an-outputstream> |
| What is InputStream & Output Stream? Why and when do we use them? | <https://stackoverflow.com/questions/1830698/what-is-inputstream-output-stream-why-and-when-do-we-use-them> |

**Java Sockets**

**1 - Contexto**

Os sockets são compostos por um conjunto de primitivas do sistema operacional e foram originalmente desenvolvidos para o BSD Unix. Podem ser utilizados nos mais variados sistemas operacionais com recursos de comunicação em rede, sendo suportados pela maioria das linguagens de programação. Sockets são suportados em Java desde o JDK 1.0. Para sua utilização devemos fazer uso das classes contidas no pacote java.net. Um exemplo interessante da programação de sockets em Java são os drivers JDBC do tipo 4, que usam sockets para comunicar-se diretamente com a API de rede do banco de dados.

Uma aplicação que utiliza sockets normalmente é composta por uma parte servidora e diversos clientes. Um cliente solicita determinado serviço ao servidor, o servidor processa a solicitação e devolve a informação ao cliente. Muitos serviços podem ser disponibilizados numa mesma máquina, sendo então diferenciados não só pelo endereço IP, mas também por um número de porta em que estão sendo ofertados.

A seguir são apresentados vídeos, leituras e fóruns de dúvidas que facilitarão seu entendimento sobre esse assunto. Você não precisa ver o conteúdo por completo, escolha os tópicos que sentir necessidade, de acordo com o tipo de mídia que você goste. Anote todas as possíveis dúvidas para levar à sala de aula no dia da revisão.

\*\*Retirado de <https://www.devmedia.com.br/java-sockets-criando-comunicacoes-em-java/9465>

**2 - Vídeos**

|  |  |
| --- | --- |
| Tema | Link |
| [PT] Socket e SocketServer (14:17) | <https://www.youtube.com/watch?v=ETfgVXk4mI8> |
| [PT] Aplicação cliente-servidor com sockets em Java - Parte 1 (14:47) | <https://www.youtube.com/watch?v=grVaGVHYnEw> |
| [PT] Sockets em Java - parte 1 (11:43) | <https://www.youtube.com/watch?v=ajEqaq-alHA> |
| [EN] Java socket programming - Simple client server program (8:58) | <https://www.youtube.com/watch?v=-xKgxqG411c> |
| [EN] Java Sockets Tutorial (9:51) | <https://www.youtube.com/watch?v=aEDV0WlwXTs> |
| [EN] Java - Sockets - Introduction (12:26) | <https://www.youtube.com/watch?v=6G_W54zuadg> |

**3 - Leituras**

|  |  |
| --- | --- |
| Tema | Link |
| Slides sobre socket | <http://www.inf.ufsc.br/~bosco.sobral/ensino/ine5645/2014-1/0050-Sockets.pdf> |
| Java Sockets: Criando comunicações em Java | <https://www.devmedia.com.br/java-sockets-criando-comunicacoes-em-java/9465> |
| Java Socket: Entendendo a classe Socket e a ServerSocket em detalhes | <https://www.devmedia.com.br/java-socket-entendendo-a-classe-socket-e-a-serversocket-em-detalhes/31894> |
| Sockets (Caelum) | <https://www.caelum.com.br/apostila-java-orientacao-objetos/apendice-sockets/#porta> |
| Documentação classe Socket - Oracle | <https://docs.oracle.com/javase/7/docs/api/java/net/Socket.html> |

**4 - Fóruns**

|  |  |
| --- | --- |
| Tema | Link |
| Sockets em java | <https://pt.stackoverflow.com/questions/42566/sockets-em-java> |
| Receber e Enviar Objetos usando Socket | <https://www.guj.com.br/t/receber-e-enviar-objetos-usando-socket/133692> |
| Conexão Socket | <https://www.guj.com.br/t/conexao-socket/334854> |
| Using Sockets to send and receive data | <https://stackoverflow.com/questions/5680259/using-sockets-to-send-and-receive-data/5680427> |
| Java Socket Programming | <https://stackoverflow.com/questions/471342/java-socket-programming> |