|  |  |
| --- | --- |
| public class MultithreadedServerSocketExample {  public static void main(String[] args) throws IOException {  try (ServerSocket server = new ServerSocket(9999)) {  while (true) {  System.out.println("Server waiting for client.");  Socket client = server.accept(); // blocks  System.out.println("Client from " + client.getLocalAddress() + " connected.");  new Thread(new RunnableWorker(client, server)).start();  }  }  }  } | public class RunnableWorker implements Runnable {  private Socket client;  private ServerSocket server;  public RunnableWorker(Socket client, ServerSocket server) {  this.client = client;  this.server = server; } @Override public void run() {  try (BufferedReader in =  new BufferedReader(new InputStreamReader(client.getInputStream()));  PrintWriter out =  new PrintWriter(new OutputStreamWriter(client.getOutputStream()))) {  // processamento da comunicação com o cliente  }  System.out.println("Client thread for socket "+client.getLocalAddress()+" is DONE!");  } } |
| Criando Servidor | |
| 1. **import** java.net.\*;  2.  3. **public class** Servidor {  4.  5. **public static void** main(String args[]) {  6. **try** {  7. *// cria um servidor*  8. ServerSocket servidor = **new** ServerSocket(18981);  9. System.out.println(“Porta 18981 aberta!”);  10.  11. *// aceita uma conexão*  12. Socket cliente = servidor.accept();  13. System.out.println(“Nova conexão com o cliente “ +  14. cliente.getInetAddress().getHostAddress()  15. );  16.  17. *// cria o buffer de leitura*  18. BufferedReader in = **new** BufferedReader(  19. **new** InputStreamReader(cliente.getInputStream())  20. );  21.  22. *// lê ate o fim*  23. **while**(**true**) {  24. String linha = in.readLine();  25. **if** (linha == **null**) {  26. **break**;  27. }  28. System.out.println(linha);  29. }  30.  31. *// fecha tudo*  32. in.close();  33. cliente.close();  34. servidor.close();  35.  36. } **catch** (IOException e) {  37.  38. *// em caso de erro*  39. System.out.println(“Ocorreu um erro na conexão”);  40. e.printStackTrace();  41. }  42. }  43. }  **Criando Cliente** |  |
| 1. **import** java.net.\*;  2.  3. **public class** Cliente {  4.  5. **public static void** main(String args[]) {  6.  7. **try** {  8. *// conecta ao servidor*  9. Socket cliente = **new** Socket(“127.0.0.1”,18981);  10. System.out.println(“O cliente se conectou ao servidor!”);  11.  12. // prepara para a leitura da linha de comando  13. BufferedReader in = **new** BufferedReader(  14. **new** InputStreamReader(System.in)  15. );  16.  17. */\* inserir o resto do programa aqui \*/*  18.  19. *// fecha tudo*  20. cliente.close();  21.  22. } **catch** (Exception e) {  23.  24. *// em caso de erro*  25. System.out.println(“Ocorreu um erro na conexão”);  26. e.printStackTrace();  27.  28. }  29. }  30. } |  |
| **Criando Servidor UDP** | ****Criando Cliente UDP**** |
| import java.io.\*;  import java.net.\*;    class UDPServer {  public static void main(String args[]) throws Exception {    int porta = 9876;  int numConn = 1;    DatagramSocket serverSocket = new DatagramSocket(porta);    byte[] receiveData = new byte[1024];  byte[] sendData = new byte[1024];    while (true) {    DatagramPacket receivePacket = new DatagramPacket(receiveData,  receiveData.length);  System.out.println("Esperando por datagrama UDP na porta " + porta);  serverSocket.receive(receivePacket);  System.out.print("Datagrama UDP [" + numConn + "] recebido...");    String sentence = new String(receivePacket.getData());  System.out.println(sentence);    InetAddress IPAddress = receivePacket.getAddress();    int port = receivePacket.getPort();    String capitalizedSentence = sentence.toUpperCase();    sendData = capitalizedSentence.getBytes();    DatagramPacket sendPacket = new DatagramPacket(sendData,  sendData.length, IPAddress, port);    System.out.print("Enviando " + capitalizedSentence + "...");    serverSocket.send(sendPacket);  System.out.println("OK\n");  }  }  } | import java.io.\*;  import java.net.\*;    class UDPClient {  public static void main(String args[]) throws Exception {    BufferedReader inFromUser = new BufferedReader(new InputStreamReader(  System.in));    DatagramSocket clientSocket = new DatagramSocket();    String servidor = "localhost";  int porta = 9876;    InetAddress IPAddress = InetAddress.getByName(servidor);    byte[] sendData = new byte[1024];  byte[] receiveData = new byte[1024];    System.out.println("Digite o texto a ser enviado ao servidor: ");  String sentence = inFromUser.readLine();  sendData = sentence.getBytes();  DatagramPacket sendPacket = new DatagramPacket(sendData,  sendData.length, IPAddress, porta);    System.out  .println("Enviando pacote UDP para " + servidor + ":" + porta);  clientSocket.send(sendPacket);    DatagramPacket receivePacket = new DatagramPacket(receiveData,  receiveData.length);    clientSocket.receive(receivePacket);  System.out.println("Pacote UDP recebido...");    String modifiedSentence = new String(receivePacket.getData());    System.out.println("Texto recebido do servidor:" + modifiedSentence);  clientSocket.close();  System.out.println("Socket cliente fechado!");  }  } |
| ****Criando Servidor Socket TCP**** | ****Criando Cliente Socket TCP**** |
| import java.io.\*;  import java.net.\*;  class TCPServer {  public static void main( String argv[]) throws Exception  {  String clientSentence;  String capitalizedSentence;  ServerSocket welcomeSocket = new ServerSocket(6789);    System.out.println("ponto1");  Socket connectionSocket = welcomeSocket.accept();    while (true) {  System.out.println("ponto2");  BufferedReader inFromClient = new BufferedReader(  new InputStreamReader(  connectionSocket.getInputStream()  )     );  DataOutputStream outToClient = new DataOutputStream(  connectionSocket.getOutputStream()  );  System.out.println("ponto3");  clientSentence = inFromClient.readLine();  System.out.println("ponto4");  capitalizedSentence = clientSentence.toUpperCase() + '\n';  outToClient.writeBytes(capitalizedSentence);  // connectionSocket.close();  }  }  } | import java.io.\*;  import java.net.\*;    class TCPClient {    public static void main(String argv[]) throws Exception {  String sentence;  String modifiedSentence;    BufferedReader inFromUser = new BufferedReader(new InputStreamReader(  System.in));    int porta = 6789;  String servidor = "localhost";    System.out.println("Conectando ao servidor " + servidor + ":" + porta);    Socket clientSocket = new Socket(servidor, porta);    DataOutputStream outToServer = new DataOutputStream(clientSocket  .getOutputStream());    BufferedReader inFromServer = new BufferedReader(new InputStreamReader(  clientSocket.getInputStream()));    System.out.println("Digite string a ser enviada para o servidor");  sentence = inFromUser.readLine();    outToServer.writeBytes(sentence + '\n');    modifiedSentence = inFromServer.readLine();    System.out.println("Recebido do servidor: " + modifiedSentence);    clientSocket.close();    }  } |