

INSTITUTO MAUÁ DE TECNOLOGIA



# Linguagens I

Programação com sockets

Profº. Tiago Sanches da Silva

# Network

# Networking Classes no JDK

Através das classes em `java.net`, os programas Java podem usar TCP ou UDP para se comunicar pela Internet. As classes **URL**, **URLConnection**, **Socket** e **ServerSocket** usam TCP para se comunicar através da rede. As classes **DatagramPacket**, **DatagramSocket** e **MulticastSocket** são para uso com UDP.



```
public class ServidorTCPBasico {  
    public static void main(String[] args) {  
        try {  
            // Instancia o ServerSocket ouvindo na porta 12345  
            ServerSocket servidor = new ServerSocket(12345);  
            System.out.println("Servidor ouvindo na porta 12345");  
            while(true) {  
                // o método accept() bloqueia até que seja recebido um pedido de conexão  
                Socket cliente = servidor.accept();  
                System.out.println("Cliente conectado: " + cliente.getInetAddress().getHostAddress());  
                ObjectOutputStream saida = new ObjectOutputStream(cliente.getOutputStream());  
                saida.flush();  
                saida.writeObject(new Object());  
                saida.close();  
                cliente.close();  
            }  
        }  
        catch(Exception e) {  
            System.out.println("Erro: " + e.getMessage());  
        }  
        finally {...}  
    }  
}
```

# Network

# Modelo OSI

Modelo de referência da ISO, tem como principal objetivo ser um modelo padrão para protocolos de comunicação entre diversos tipos de sistema, garantindo a comunicação **end-to-end**, o Modelo **OSI** (em inglês Open Systems Interconnection) foi lançado em 1984 pela Organização Internacional para a Normalização (em inglês International Organization for Standardization).

# Modelo OSI

Trata-se de uma arquitetura modelo que divide as redes de computadores em 7 camadas que possuem responsabilidades distintas com o intuito de criar um meio possível para comunicação.

Cada protocolo realiza a inserção de uma funcionalidade assinalada a uma camada específica.



# Modelo OSI

O Modelo OSI é composto por 7 camadas, sendo que cada uma delas realizam determinadas funções. As camadas são: **Aplicação** (Application), **Apresentação** (Presentation), **Sessão** (Session), **Transporte** (Transport), **Rede** (Network), **Dados** (Data Link) e **Física** (Physical).

Ex. protocolos em cada layer: [https://pt.wikipedia.org/wiki/Modelo\\_OSI](https://pt.wikipedia.org/wiki/Modelo_OSI)

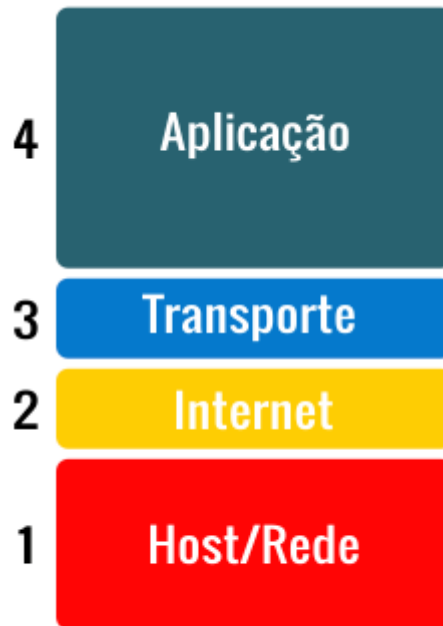


Modelo de Referência OSI

# Modelo TCP/IP

A arquitetura TCP/IP assim como a OSI, possui suas funções divididas em camadas, com a diferença na quantidade, enquanto o Modelo OSI possui 7 camadas, o Modelo TCP/IP possui apenas 4 camadas.

Resumo das camadas: <https://www.uniaogeek.com.br/arquitetura-de-redes-tcpip/>



Modelo de Referência TCP/IP

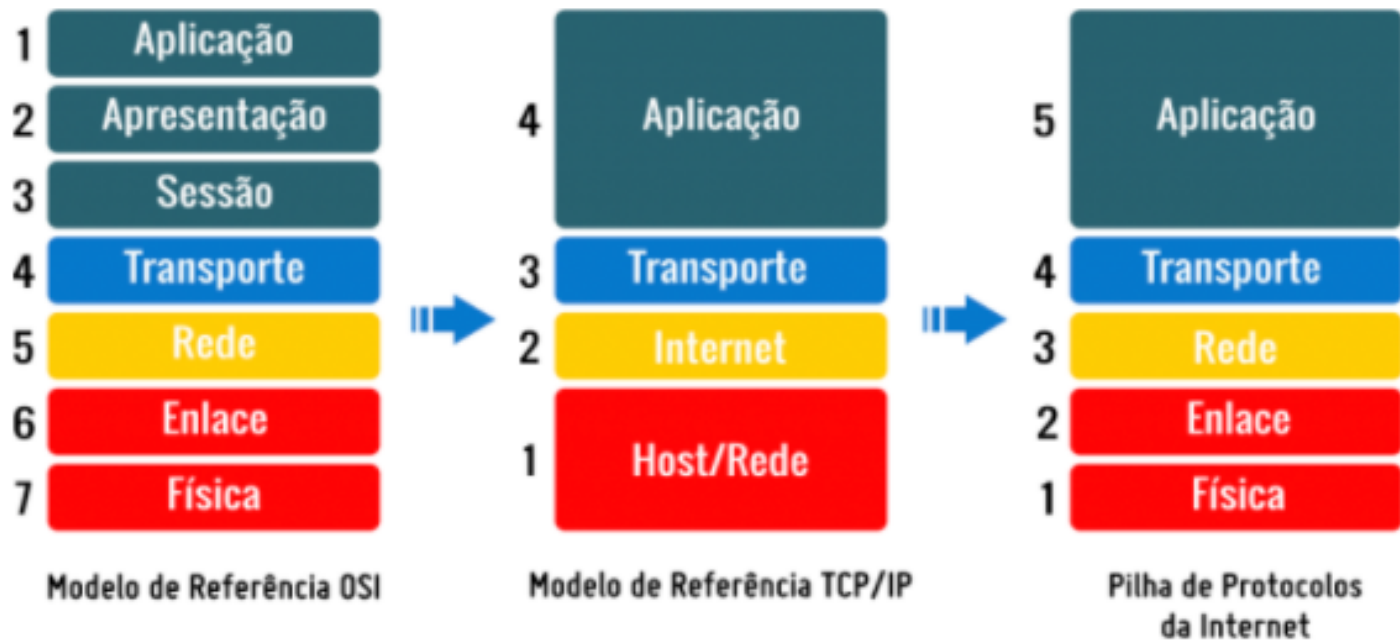


# **A internet como conhecemos**

# Modelo TCP/IP

A internet utiliza uma pilha de protocolos mista, ou seja, ela é resultado da mistura das duas pilhas de protocolos (OSI e TCP/IP). O resultado desta mistura, gerou um Modelo de abstração em 5 camadas, conforme indicado abaixo:

**Modelo OSI + Modelo TCP/IP = Pilhas de Protocolos da Internet**



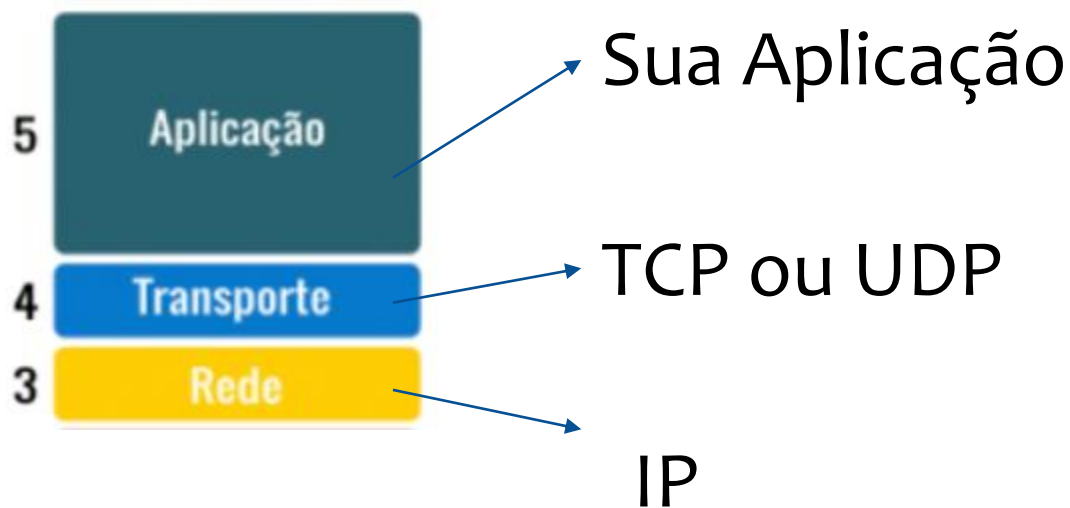
# Modelo TCP/IP

- Camada de Aplicação
  - Suporta as aplicações da rede.
    - Ex.: FTP, SMTP, HTTP...
- Camada de Transporte
  - Transferência de dados, sistema final a sistema final. Ex.: **TCP** e **UDP**
- Camada de Rede
  - Roteamento de **datagramas** da origem ao destino. Ex.: **IP**, protocolos de roteamento.
- Camada de Enlace
  - Transferência de dados entre elementos vizinhos da rede. Ex.: PPP, Ethernet...
- Camada Física
  - Bits no meio de transmissão. Ex.: Pulsos elétricos no Cabo UTP.



# O que nos importa agora?

Não se preocupe agora com entender tudo sobre protocolos, você precisar saber apenas como os **sockets** funcionam, então vamos lá.

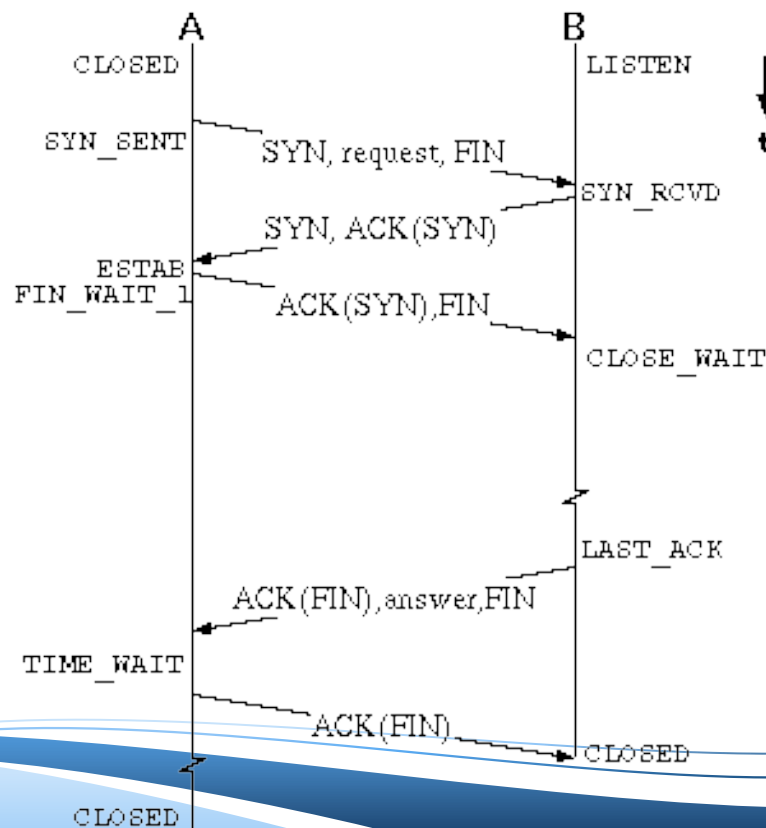


**TCP ou UDP?**

# TCP x UDP (RESUMIDAMENTISÍSSIMO)

O TCP é o protocolo mais usado isto porque fornece garantia na entrega de todos os pacotes entre um PC emissor e um PC receptor.

No estabelecimento de ligação entre emissor e receptor existe um “pré-acordo” denominado de Three Way Handshake (SYN, SYN-ACK, ACK).



# TCP x UDP (RESUMIDAMENTISÍSSIMO)

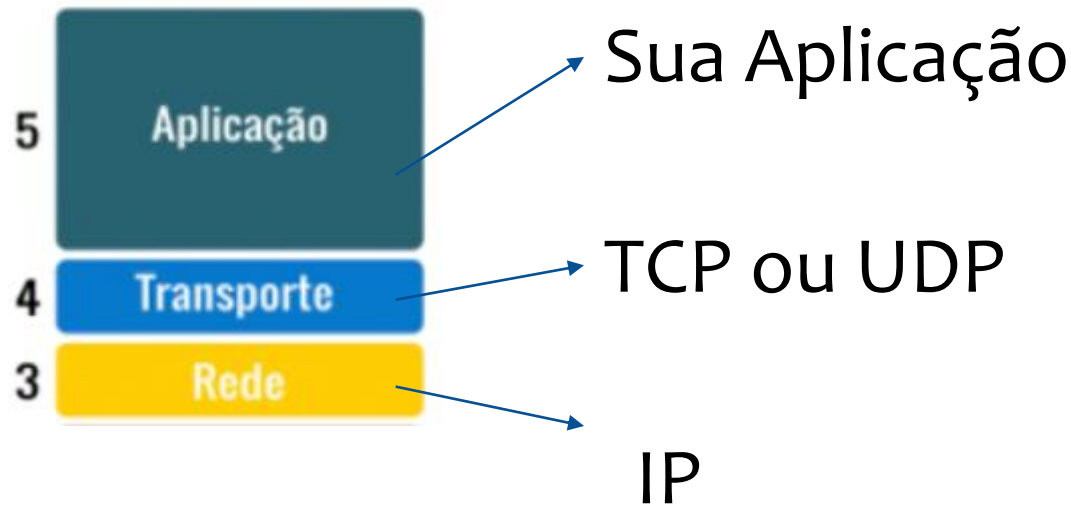
UDP é um protocolo mais simples e por si só não fornece garantia na entrega dos pacotes. No entanto, esse processo de garantia de dados pode ser simplesmente realizado pela aplicação em si (que usa o protocolo UDP) e não pelo protocolo. Basicamente, usando UDP, uma máquina emissor envia uma determinada informação e a máquina receptor recebe essa informação, não existindo qualquer confirmação dos pacotes recebidos. Se um pacote se perder não existe normalmente solicitação de reenvio, simplesmente não existe.

[http://www.diffen.com/difference/TCP\\_vs\\_UDP](http://www.diffen.com/difference/TCP_vs_UDP)

- Menor overhead
- Desempenho
- Baixa latência
- Para que ele pode ser usado?

# O que nos importa mesmo?

Saber apenas como os **sockets** funcionam.



MAS...

De um modo geral, um computador possui uma única conexão física com a rede. Todos os dados destinados a um computador específico chegam através dessa conexão. No entanto, os dados podem estar destinados a diferentes aplicativos em execução no computador. Então, como o computador conhece o aplicativo para encaminhar os dados? PORTAS!



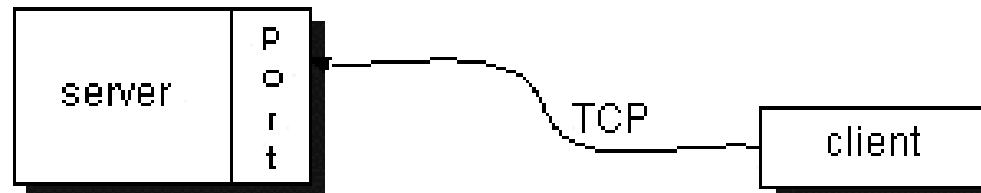
# **Entendendo as portas**

# Portas

Os dados transmitidos pela Internet são acompanhados por informações de endereçamento que identificam o computador e a porta para o qual está destinada.

O computador é identificado pelo seu endereço IP de 32 bits, que o IP usa para fornecer dados para o computador certo na rede. As portas são identificadas por um número de 16 bits, que TCP e UDP usam para entregar os dados para o aplicativo correto.

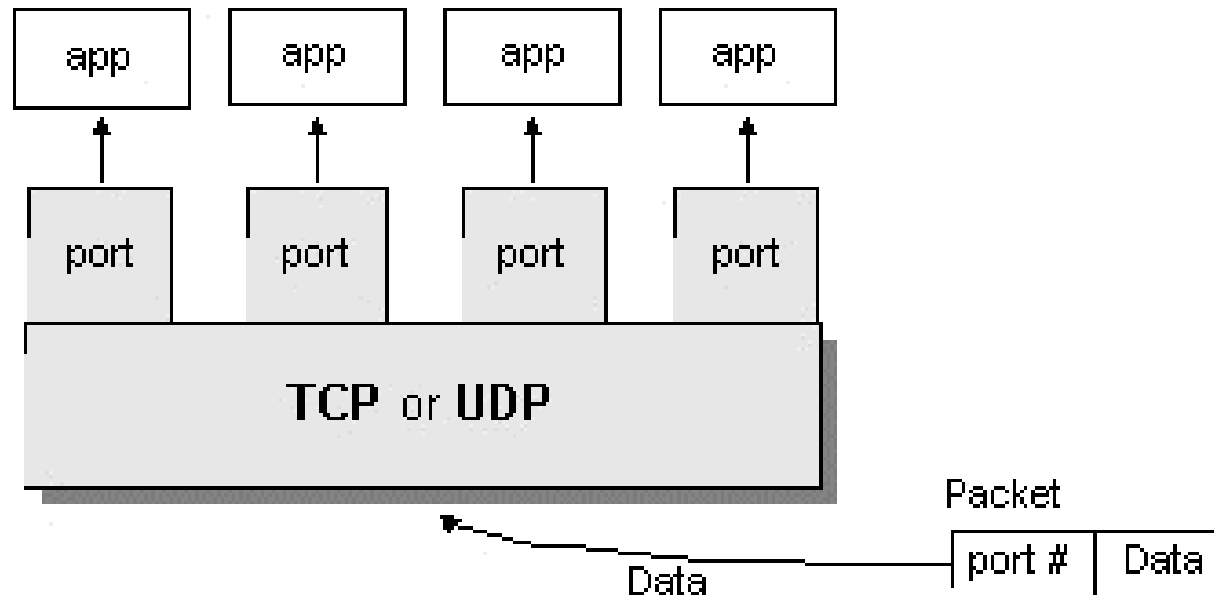
Em uma comunicação baseada em conexão, como TCP, um aplicativo de servidor vincula um **socket** a um número de porta específico.



# Portas

Definição:

Os protocolos TCP e UDP usam portas para mapear dados recebidos para um processo específico em execução em um computador.



# Que número de porta utilizar?

Os números de portas variam de 0 a 65535 porque as portas são representadas por números de 16 bits. Os números das portas que variam de 0 a 1023 são restritos; eles são reservados para uso por serviços bem conhecidos, como HTTP e FTP e outros serviços do sistema. Essas portas são chamadas de “well-known ports”. Suas aplicações não devem tentar se ligar a elas.

<https://www.iana.org/>

<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>



# Sockets

# O que são sockets?

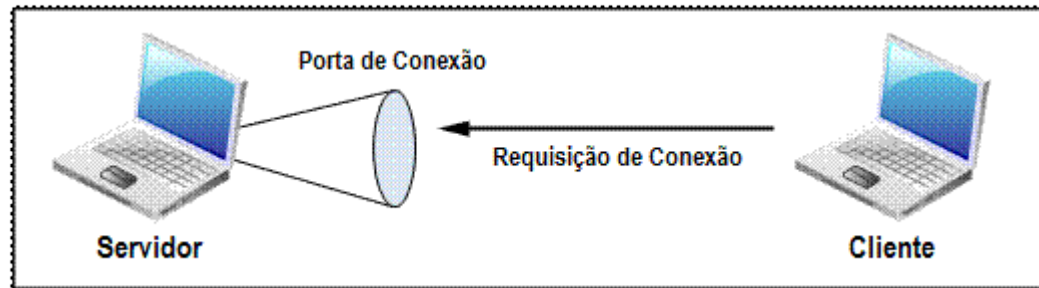
Socket ou soquete é apenas um conceito ou uma abstração. O termo socket é utilizado para representar um ponto de conexão para uma rede de computadores que utiliza o protocolo TCP/IP.

Quando dois computadores necessitam manter uma comunicação, cada um deles utiliza um socket.



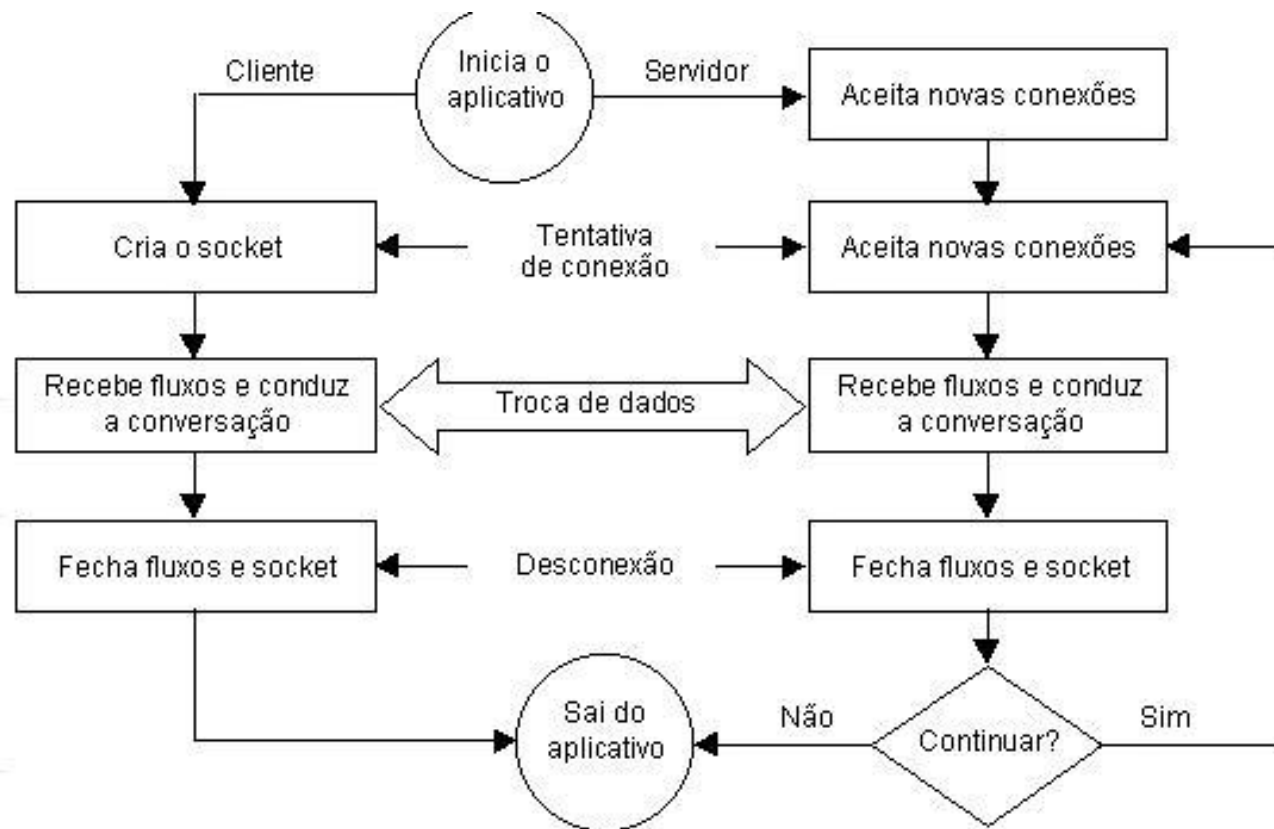
# Cliente-Servidor

Um computador, denominado server ou servidor, disponibiliza um socket e aguarda o recebimento de uma solicitação de conexão, enquanto outro computador, denominado client ou cliente, executa um socket para se comunicar à máquina servidora.



# Cliente-Servidor

Uma aplicação que utiliza sockets normalmente é composta por uma parte servidora e diversos clientes.








**Vamos ao java logo?**

# Networking Classes no JDK

Através das classes em `java.net`, os programas Java podem usar TCP ou UDP para se comunicar pela Internet. As classes **URL**, **URLConnection**, **Socket** e **ServerSocket** usam TCP para se comunicar através da rede. As classes **DatagramPacket**, **DatagramSocket** e **MulticastSocket** são para uso com UDP.



# Pesquisa

- ServerSocket
  - O que é?
  - Como funciona o .accept?
- Socket
- OutputStream (getOutputStream)
  - O que é, para que serve e como utilizar;
- InputStrem (getInputStream)
  - O que é, para que serve e como utilizar;
- Explique como funciona o código de exemplo Servidor e Cliente básico fornecido.

# Código Servidor exemplo

```
public class ServidorTCPBasico {
    public static void main(String[] args) {
        try {
            // Instancia o ServerSocket ouvindo a porta 12345
            ServerSocket servidor = new ServerSocket(12345);
            System.out.println("Servidor ouvindo a porta 12345");
            while(true) {
                // o método accept() bloqueia a execução até que
                // o servidor receba um pedido de conexão
                Socket cliente = servidor.accept();
                System.out.println("Cliente conectado: " + cliente.getInetAddress().getHostAddress());
                ObjectOutputStream saida = new ObjectOutputStream(cliente.getOutputStream());
                saida.flush();
                saida.writeObject(new Date());
                saida.close();
                cliente.close();
            }
        }
        catch(Exception e) {
            System.out.println("Erro: " + e.getMessage());
        }
        finally {...}
    }
}
```

# Código Cliente exemplo

```
public class ClienteTCPBasico {  
    public static void main(String[] args) {  
        try {  
            Socket cliente = new Socket("paulo",12345);  
            ObjectInputStream entrada = new ObjectInputStream(cliente.getInputStream());  
            Date data_atual = (Date)entrada.readObject();  
            System.out.println("Data recebida do servidor:" + data_atual.toString());  
            entrada.close();  
            System.out.println("Conexão encerrada");  
        }  
        catch(Exception e) {  
            System.out.println("Erro: " + e.getMessage());  
        }  
    }  
}
```

**Próxima aula tem mais**

Perguntas?

# Referências

- <https://www.uniaogeek.com.br/arquitetura-de-redes-tcpip/>
- <https://docs.oracle.com/javase/tutorial/networking/overview/networking.html>
- Deitel
- <http://www.devmedia.com.br/programacao-de-sockets-em-java/21138>
- <https://canaltech.com.br/produtos/o-que-e-modelo-osi/>
- <http://www.devmedia.com.br/java-sockets-criando-comunicacoes-em-java/9465>