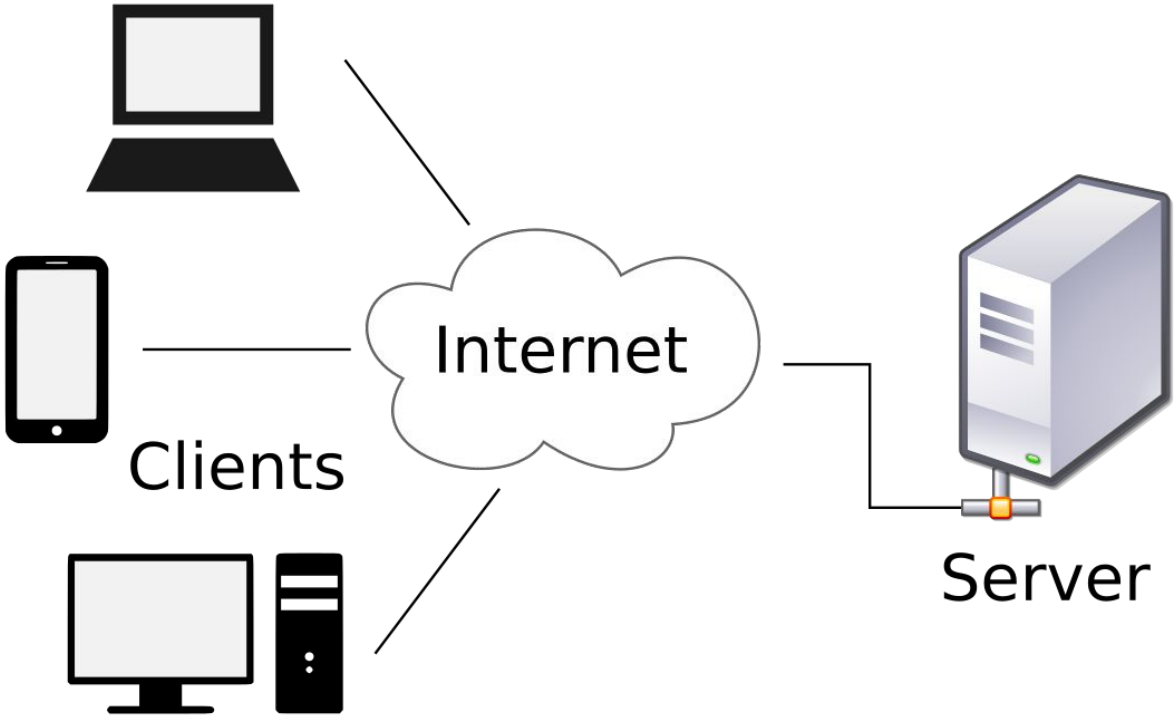
Desenvolvimento de Software para Dispositivos Móveis

Aula 10 - Socket

Professor: Anderson Almada

Socket



Socket

- Server
  - Você deve definir em que porta será executado o serviço oferecido
    - **ServerSocket server = new ServerSocket(5560);**
  - Definido a porta, o serviço deve ficar constantemente recebendo as requisições, para isso é feito um laço infinito
    - **while(true) {**

      }

Socket

- Server
  - Dentro do laço, chame o método responsável por aceitar a conexão
    - **Socket socket = server.accept();**
  - Depois desse método, o socket está aberto para as comunicações do serviço. É possível tanto enviar como receber dados.
    - Com a classe **DataInputStream** é possível recuperar dados da entrada
    - Com a classe **DataOutputStream** é possível enviar dados para saída

Socket

- Server
  - O socket possui os dois fluxos (entrada/saída). Assim:

```
DataInputStream dIn = new DataInputStream(socket.getInputStream());

DataOutputStream dOut = new DataOutputStream(socket.getOutputStream());
```

Socket

- Server
  - Se caso seja necessário recuperar dados, só utilizar o objeto **dIn:**

String -> dIn.readUTF()

int -> dIn.readInt()

byte[] -> dIn.read();

Socket

- Server
  - Se caso seja necessário enviar dados, só utilizar o objeto **dOut:**

dOut.writeUTF(“String”) -> String

dOut.writeInt(20) -> int

dOut.write(bytes); -> byte[]

Socket

- Server
  - Quando terminar de enviar e receber todos os dados necessários, feche todos os fluxos e o socket

```
dOut.flush();
dOut.close();
dIn.close();
socket.close();
```

Socket

- Server
  - https://pastebin.com/LdUJjVsG

Socket

- Client
  - A solicitação do serviço deve ser executado dentro de uma Thread
  - Você deve apontar para o endereço do serviço solicitado (ip/porta)
    - **Socket socket = new Socket("ip", 5560);**

Socket

- Client
  - Da mesma forma que o servidor, o cliente do serviço pode utilizar as classes **DataInputStream** e **DataOutputStream.** Assim:

```
DataInputStream dIn = new DataInputStream(socket.getInputStream());

DataOutputStream dOut = new DataOutputStream(socket.getOutputStream());
```

Socket

- Client
  - Lembrar que se o servidor envia um **dOut** com dados, o cliente precisa ter um **dIn** para recuperar os dados.

  - Da mesma forma quando o cliente envia um **dOut**, precisa ter um servidor com **dIn** para receber.

Socket

- Client
  - https://pastebin.com/BuRQT1yN

Exercício

- Acesse o serviço que roda na porta 60000 do servidor
  - ip: 192.168.0.114

- O serviço retorna o timestamp (Long) do servidor

- Crie um botão na aplicação que a cada clique, ele solicita esse valor do timestamp, converte em uma data e armazena em uma lista (recyclerView)

Exercício

- https://developer.android.com/reference/java/net/Socket

- https://www.techemportugues.com/2017/08/12/programar-um-servidor-java-um-cliente-android-usando-sockets/

Dúvidas??

E-mail: almada@crateus.ufc.br