15-440 Distributed Systems Recitation 2

Tamim Jabban

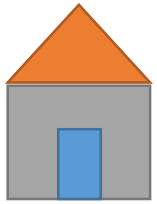
Project 1

- Involves creating a *Distributed File System* (DFS)
- Released yesterday
- When/If done with PS1, start reading the handout
- **Today**: Socket communication!

Communication via Sockets

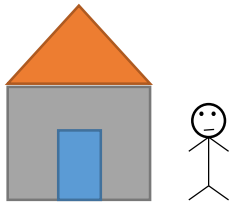
- Sockets provide a communication mechanism between networked computers.
- A Socket is an end-point of communication that is identified by an IP address and port number.
- A client sends requests to a server using a client socket.
- A server receives clients' requests via a listening socket

Communication via Sockets



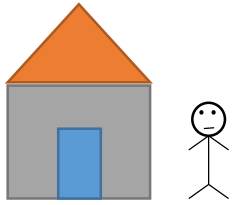
Person A
(A's home)

Person B
(Guest)



Person B
Knocks the door

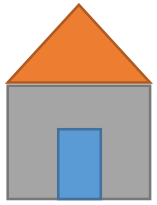
Person A
Is Listening



Person B
Enters

Person A
Opens door

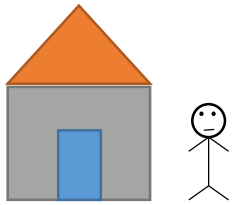
Communication via Sockets



A "binds" to his home

Person A
(A's home)

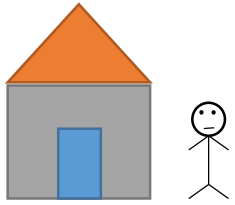
Person B
(Guest)



Person B
Knocks the door

B sends a request to communicate

Person A
Is Listening



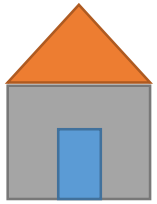
A accepts the request

Person B
Enters

B is now "connected" with A

Person A
Opens door

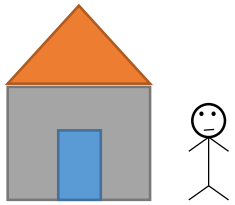
Communication via Sockets



Server A

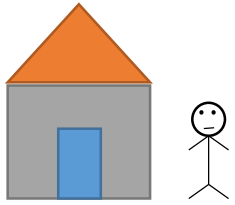
A binds to:
(1) IP address
(2) Port number

Client B



Server A is Listening to Requests

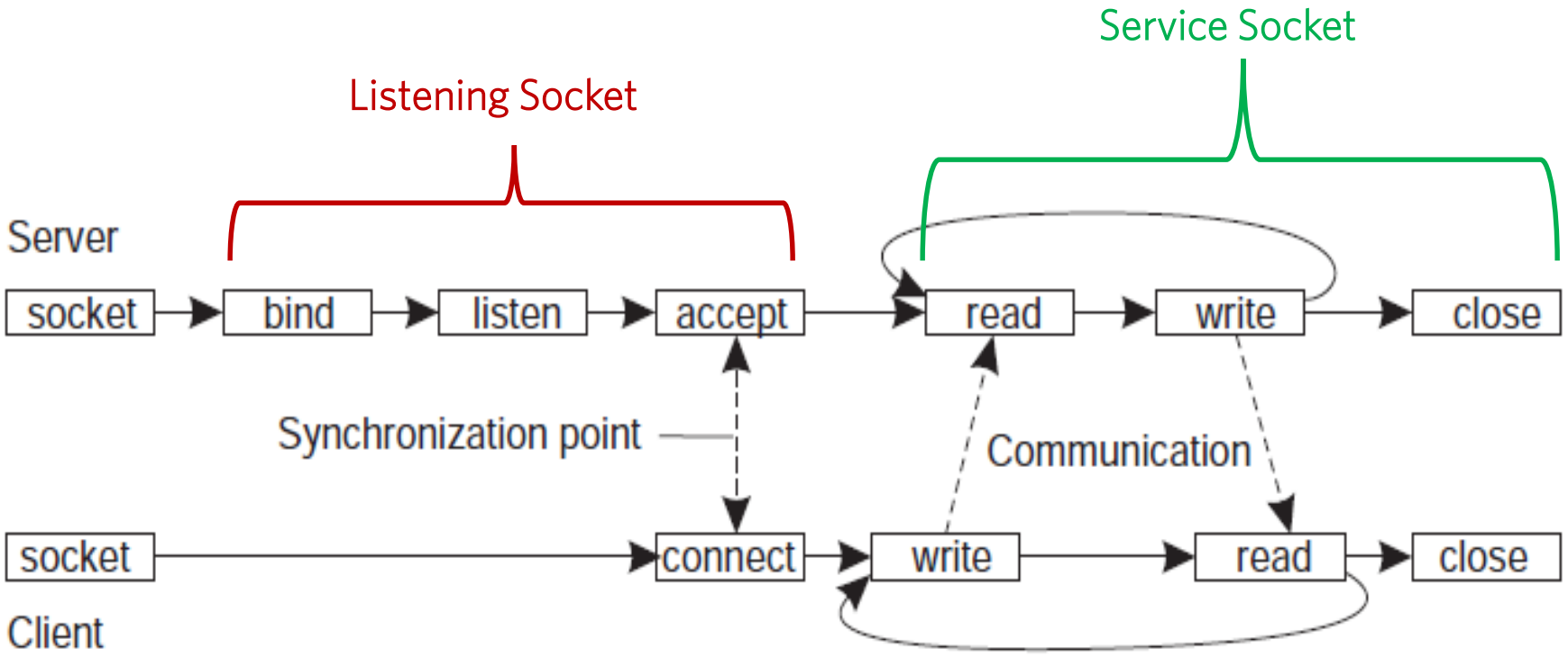
Client B sends a request to communicate with the server



Server A accepts request

Client B is now connected with Server A

Communication via Sockets



Socket Communication Recipe

1. Server instantiates a ServerSocket object (usually passing a port number). This socket is referred to as the listening socket.

2. Server invokes the **`accept()`** method that awaits incoming client connections.

3. Client instantiates Socket object (passing server name and/or port number). This socket is referred to as a client socket*.

4. On the server side, **`accept()`** returns a new socket referred to as a service socket on which the client reads/writes.

* The constructor of the Socket class attempts to connect the client to the specified server and port number. If communication is established, the client now has a Socket object capable of communicating with the server.

ServerSocket Methods

SN	Methods with Description
1	**public ServerSocket(int port) throws IOException** Attempts to create a server socket bound to the specified port. An exception occurs if the port is already bound by another application.
2	**public ServerSocket() throws IOException** Creates an unbound server socket. When using this constructor, use the bind() method when you are ready to bind the server socket.
3	**public Socket accept() throws IOException** Waits for an incoming client. This method blocks until either a client connects to the server on the specified port or the socket times out, assuming that the time-out value has been set using the setSoTimeout() method. Otherwise, this method blocks indefinitely.
4	**public void bind(SocketAddress host)** Binds the socket to the specified server and port in the SocketAddress object. Use this method if you instantiated the ServerSocket using the no-argument constructor.
5	**public SocketAddress getLocalSocketAddress()** Returns the address of the endpoint this socket is bound to, or null if it not bound yet.

Socket Methods

SN	Methods with Description
1	**public Socket(String host, int port) throws UnknownHostException, IOException** This method attempts to connect to the specified server at the specified port. If this constructor does not throw an exception, the connection is successful and the client is connected to the server.
2	**public Socket()** Creates an unconnected socket. Use the connect() method to connect this socket to a server.
3	**public void connect(SocketAddress host, int timeout) throws IOException** This method connects the socket to the specified host. This method is needed only when you instantiated the Socket using the no-argument constructor.
4	**public InputStream getInputStream() throws IOException** Returns the input stream of the socket. The input stream is connected to the output stream of the remote socket.
5	**public OutputStream getOutputStream() throws IOException** Returns the output stream of the socket. The output stream is connected to the input stream of the remote socket
6	**public SocketAddress getLocalSocketAddress() throws IOException** Returns the address of the endpoint this socket is bound to, or null if it is not bound yet.
7	**public void close() throws IOException** Closes the socket, which makes this Socket object no longer capable of connecting again to any server

InetSocketAddress Methods

SN	Methods with Description
1	**public InetSocketAddress(String hostname, int port)** Creates a socket address from a hostname and a port number.
2	**public InetSocketAddress(String hostname, int port)** Creates a socket address from a hostname and a port number.
3	**public InputStream getInputStream() throws IOException** Returns the input stream of the socket. The input stream is connected to the output stream of the remote socket.
4	**public OutputStream getOutputStream() throws IOException** Returns the output stream of the socket. The output stream is connected to the input stream of the remote socket
5	**public void close() throws IOException** Closes the socket, which makes this Socket object no longer capable of connecting again to any server

Multi-Threading

- A class intended to execute as a thread must implement the ***Runnable*** interface

```
public class Service implements Runnable
```

- STEP 1: Implement the method ***run()***

```
public void run() {  //thread's logic goes here }
```

- STEP 2: Instantiate a Thread object

```
Thread t = new Thread(new Service())
```

- STEP 5: Invoke ***start()*** on the new thread

```
t.start()                    // invokes the run() method
```

Transport Protocols

- Socket: endpoint to read and write data
- Each Socket has a **network protocol**
- Two types of **protocols** used for communicating data/*packets* over the internet:
  - TCP:
    - ***Transmission Control Protocol***
    - Connection Oriented (*handshake*)
  - UDP:
    - ***User Datagram Protocol***
    - "Connectionless"