43320

Page :
Date : / /

Assignment no 1

Title : Distributed application using Java Socket and RMI

Objectives

1] Upon successfully completion of this course, Student should be able to demonstrate knowledge of the basic element

Problem Statement :

To develop any distributed application through implementing client-server communication program based on java socket

Software / hardware requirement

* Software => 1] Ubuntu 16.04
* Hardware => 2] Dual core / Quad core machine with 4GB Ram

Theory

Java Socket

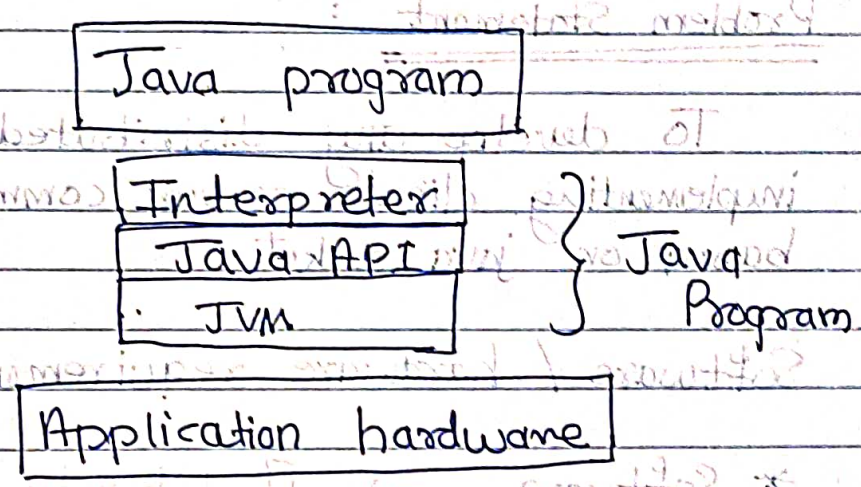
* Socket provide the communication mechanism between 2 computers using TCP
* A client program creates a socket on its end of communication & attempts to connect that socket to a server.

43320

Page:
Date: / /

Java API

* The Java API is the set of classes included with the Java development environment
* These classes are written using the java language and run on the JVM.
* The following image depicts the fundamental components of Java API

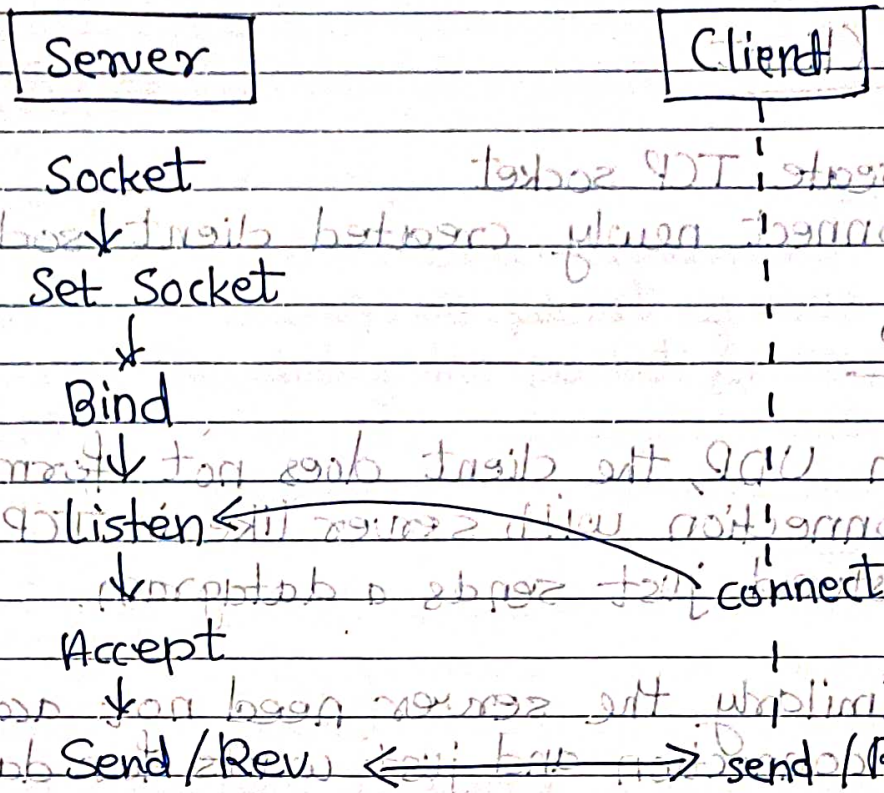


TCP

* TCP [Transmission Control Protocol] is a connection oriented communication
* It is an intermediate layer of the application layer and internet protocol layer in OSI model.

13320

Page :
Date : / /



TCP Server

1] Using create(), Create TCP Socket.

2] Using bind(), Bind the socket to server.

3] Using listen(), put the server socket in a passive mode, where it waits for the client to approach the server to make a connection

4] Using accept(), at this point, connection is established between client & server, they are ready to transfer data.

5] Goto step 3

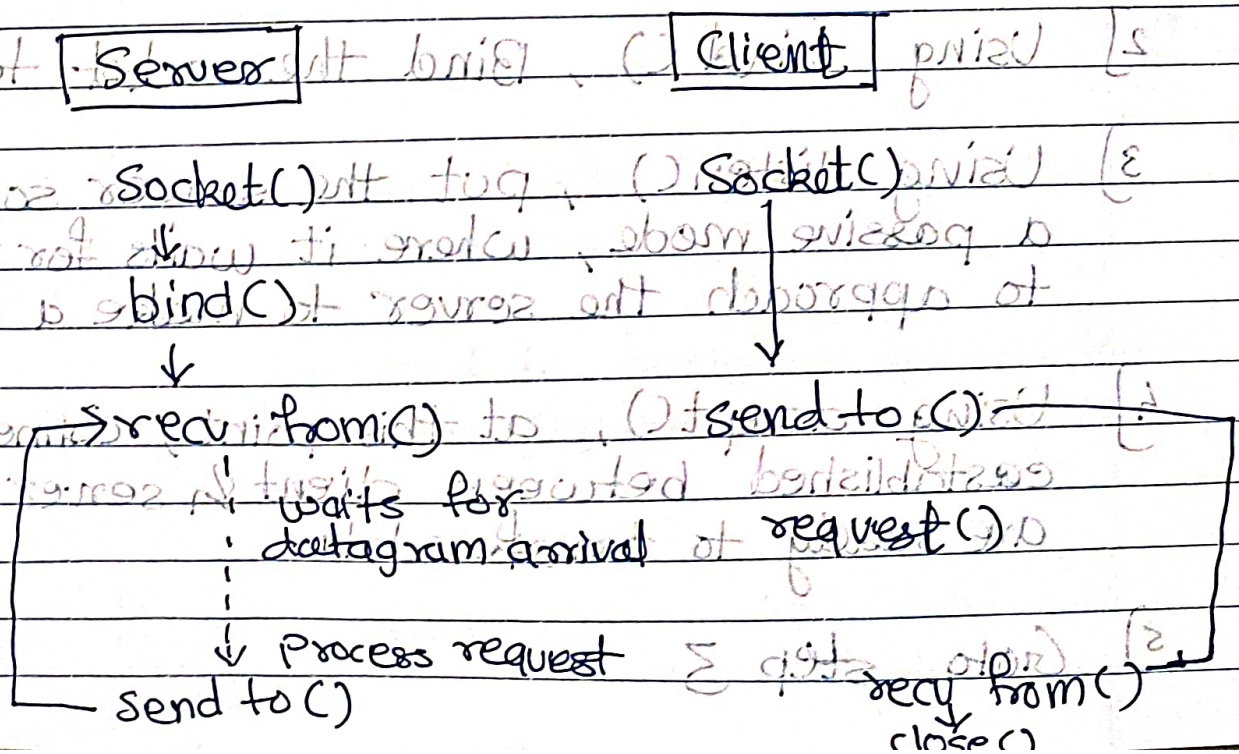
Page :
Date : / /

TCP Client

1) Create TCP socket
2) Connect newly created client socket to server

UDP

* In UDP, the client does not form a connection with server like in TCP instead just sends a datagram.
* Similarly the server need not accept a connection and just waits for datagram to arrive
* Datagram upon arrival contains the address of sender, which the server uses to send data to the current client



43320

Page :
Date : / /

UDP Server

1) Create UDP Client
2) Bind the socket to server address
3) Wait until datagram packet arrives from client
4) Process the datagram packet & send a reply to client
5) Go back to step 3

UDP Client

1) Create UP socket
2) Send message to server
3) Wait until datagram packet arrives from client.
4) Process the datagram packet & send a reply to client
5) Go back to step 3
6) Close socket, exit

Conclusion

In this way, we have developed a distributed application through implementing client server communication program based on java socket.

43320

Assignment no : 1B

Page :
Date : / /

Theory -

RMI

RMI is an API which allows an object to invoke a method of an object that exists in another address space, which could be on the same machine or on a remote machine.

Through RMI, object running in a JVM present on a computer [client-side] can invoke method on a object present in another JVM.

RMI Client server communication is handled by 2 intermediate object.

1] Stub object [on Client side]

It builds an information block & sends this information to the server

The block is consists of

(1) An identifier of remote object to be used
(2) Method name which is to be invoked
(3) Parameters to the remote JVM.

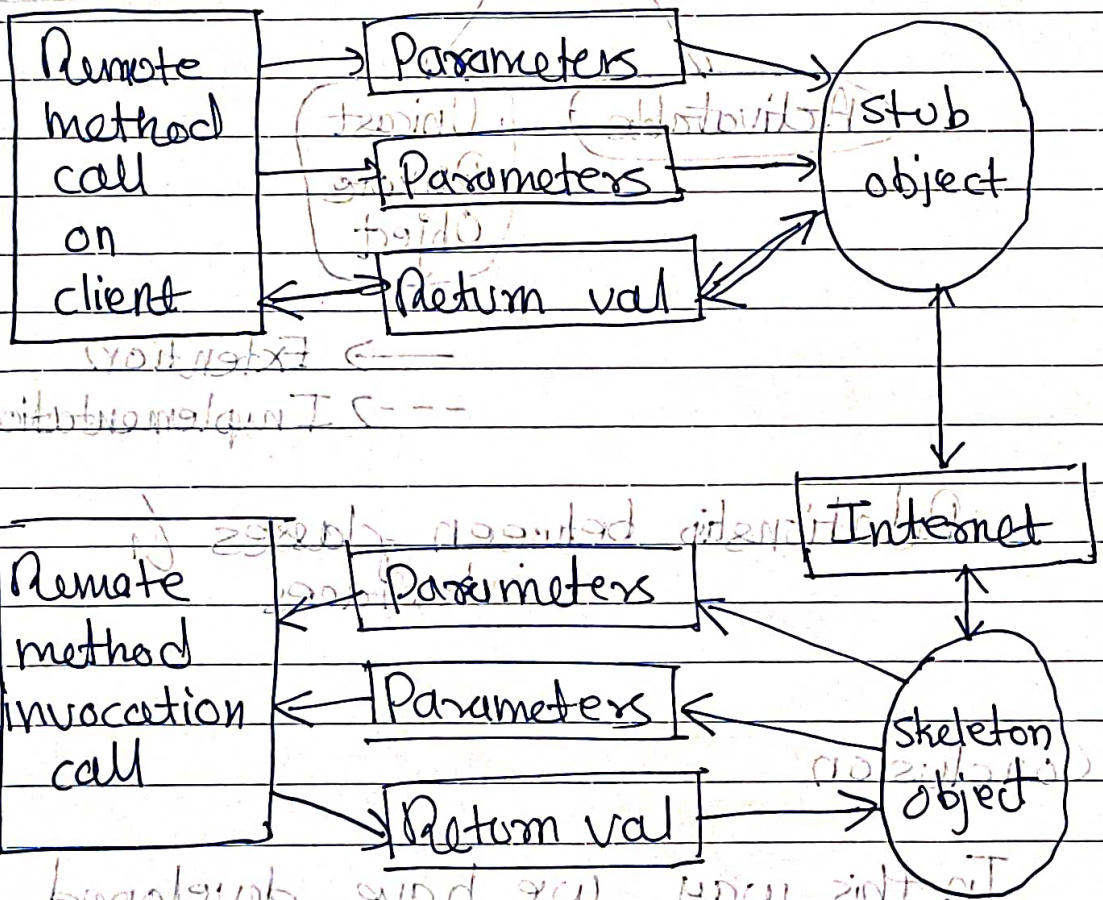
Page :
Date : / /

2] Skeleton Object [on server side]

It passes the request from stub object to remote object.

(1) It calls the parameter received from the stub object to the method.

(2) It forwards the parameters received from stub to the method.

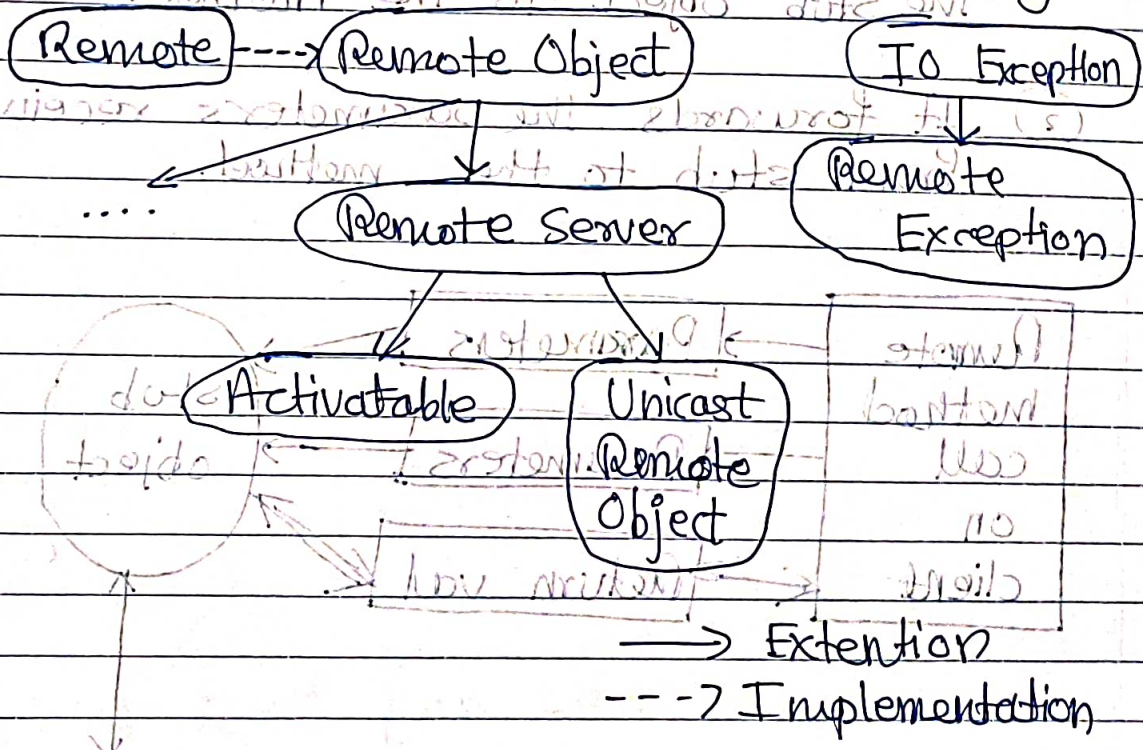


Working of RMI

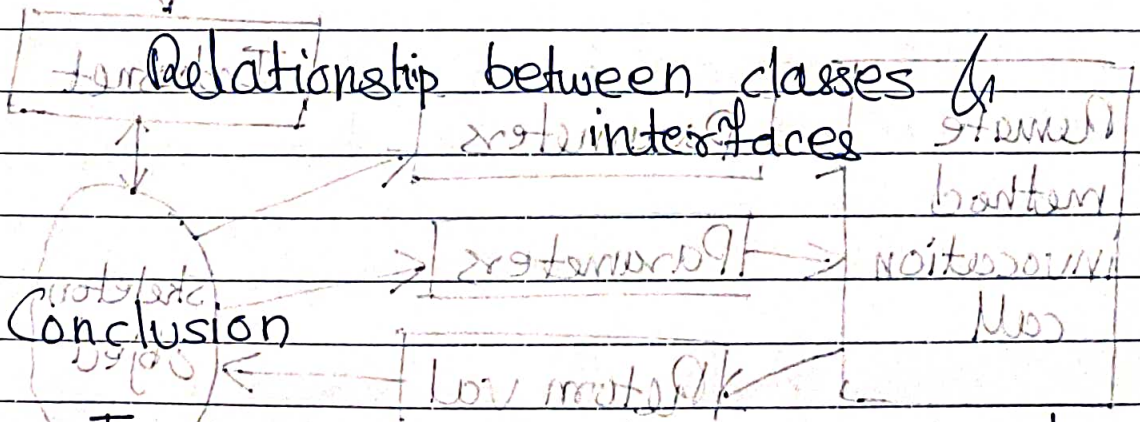
Page :
Date : / /

Interfaces & classes.

Interfaces & classes that are responsible for specifying the remote behaviour of the RMI system are defined in the java.rmi. hierarchy



Relationship between classes & interfaces



Conclusion

In this way, we have developed distributed application through implementing client server communication based on Java RMI.